NASA Contractor Report 191054

Hierarchical Poly Tree Configurations for the Solution of Dynamically Refined Finite Element Models

G.D. Gute and J. Padovan
University of Akron
Akron, Ohio

January 1993

Prepared for
Lewis Research Center
Under Grant NAG3–664

Hierarchical Poly Tree Configurations for the Solution of Dynamically Refined Finite Element Models

G. D. Gute and J. Padovan
University of Akron
Akron, Ohio 44325

Abstract

This paper demonstrates how a multilevel substructuring technique, called the Hierarchical Poly Tree (HPT), can be used to integrate a localized mesh refinement into the original finite element model more efficiently. The optimal HPT configurations for solving isoparametrically square *h*-, *p*-, and *hp*-extensions on single and multiprocessor computers is derived. In addition, the reduced number of stiffness matrix elements that must be stored when employing this type of solution strategy is quantified. Moreover, the HPT inherently provides localized "error-trapping" and a logical, efficient means with which to isolate physically anomalous and analytically singular behavior.

1 INTRODUCTION

With the advent of affordable computer resources, engineers have come to rely upon numerical techniques to simulate various types of physical phenomena. These include structural mechanics, fluid dynamics, electromagnetic fields, and heat transfer to name a few. The most widely used methods for numerically approximating this behavior are generalized Finite Element (FE) and Finite Difference (FD) formulations.

Due to the limited approximation/interpolation capabilities of the aforementioned numerical techniques, the results of an analysis depend heavily upon the proper discretization of the system in question. Oftentimes, as the solution process proceeds, various localized phenomena may occur that would require a refinement of the model to ensure the reliability of the solution. Such model refinements are triggered by the occurrence of;

1. Shock wave formation,
2. Cracking,
3. Material nonlinearity,
4. Geometric nonlinearity,
5. Boundary layer formation, and
6. Varying boundary conditions, etc.

To date, many mesh refinement schemes have been developed that address this issue. These schemes are typically classified as r-, h-, or p-extensions; i.e.,

r-extension: Is a node relocation scheme which adapts the spatial coordinates of the nodes toward the optimal location. The number of nodes and elements are fixed when using this approach;

h-extension: Is a scheme wherein elements containing a large amount of error are refined into much smaller elements;

p-extension: Is a method that employs higher order polynomials for the shape functions of elements containing a large amount of finite element approximation error.

These schemes have been successfully used, both individually and concomitantly, to solve PDEs of the elliptical, parabolic, and hyperbolic types in one and two dimensions[1-3]. Due to the large expanse of literature available that pertains to this subject it would be impractical, if not impossible, to reference all of the authors that have made contributions related to the development of these techniques. In light of this, we will simply refer the reader to the cumulative works of I. Babuška and B.A. Szabó as well as the publications compiled in [4].

Regardless of whether an *r*-, *h*-, and/or *p*-extension is employed, several iterations of the solution process are required before satisfactory results can be attained. It directly follows that the implementation of these techniques is somewhat restrictive, especially for large FE systems, because of the large computational costs involved. In this context, we will illustrate how the Hierarchical Poly Tree (HPT)[5,6] solution strategy can be incorporated into the aforementioned mesh refinement routines so as to yield more efficient computational algorithms for both sequential and multiprocessor type machines. In addition, an HPT inherently provides;

1. The minimization of in-core and out-of-core memory requirements;
2. A logical/efficient means of "isolating" localized mesh adaptations;
3. Localized "error-trapping", i.e., the influence of localized modelling errors are essentially confined to their "branch" of the Tree;
4. An orderly multilevel organization of the model topology wherein interpolative reduction schemes can be employed between levels for simulations involving a hierarchy of fine to very coarse scales in its definition[5,6].

In this context, the forthcoming chapters will:

1. Provide the motivation/philosophy of utilizing the HPT for solving dynamically refined FE discretizations,

2. Develop the optimal multilevel HPT for the solution of locally refined regions in a sequential type computing environment,

3. Illustrate the potential advantages of using the HPT solution strategy in a multiprocessor environment.

2 HPT PHILOSOPHY

The Hierarchical Poly Tree (HPT) is a multilevel substructuring technique that has been shown to yield significant speed enhancements with reqards to the solution of the resulting system of simultaneous algebraic equations that arise from FE formulations. Moreover, when implemented in a multiprocessor environment, the HPT has the potential of solving the numerical problem with superlinear speed enhancements, i.e. the resultant speedup is greater than the number of processors used in parallel. Given that the use of the aforementioned dynamic mesh refinement schemes typically necessitates the solution of the system equations several times before an adequate solution is attained, the incorporation of the HPT solution strategy into their respective algorithms would prove to be very advantageous. This is especially true when addressing large FE systems.

The development of the HPT solution strategy evolved from the recognition that the computational effort associated with the solution of

$$[K]\{Y\} = \{F\} \tag{2.1}$$

can be approximated as [7],

$$C_g = \frac{1}{2} N_E [N_B(N_B + 1)] + 2 N_E N_B \tag{2.2}$$

where

C_g – total number of arithmetic operations necessary to solve Equation (2.1)

N_E – number of equations/unknowns

N_B – mean-half-bandwidth of $[K]$

$\frac{1}{2} N_E [N_B(N_B + 1)]$ – number of arithmetic operations required to perform an $[L][D][L]^T$ factorization of $[K]$ via a direct, skylined solver such as the one employed in ADINA [7,8]

$2 N_E N_B$ – number of arithmetic operations needed to factorize $\{F\}$ and perform the subsequent back substitution step for calculating $\{Y\}$.

Furthermore, since $[K]$ is symmetric, the mean-half-bandwidth is defined to be

$$N_B = \frac{\beta}{\alpha} \tag{2.3}$$

where

$\alpha \equiv N_E$

β – number of stiffness matrix elements stored in the upper triangular of $[K]$ via a skylined method.

Substituting (2.3) into (2.2) yields

$$\begin{aligned} C_g &= \frac{\beta_g}{2}\left(\frac{\beta_g}{\alpha_g} + 5\right) \\ &\sim \frac{1}{2}\frac{(\beta_g)^2}{\alpha_g} \end{aligned} \tag{2.4}$$

Thus, equation (2.4) is an approximation of the computational effort associated with the solution of (2.1) in terms of the number of equations and stiffness matrix elements.

Now, consider the process of hierarchical substructuring as shown in Figure 2.1. Note that the 1^{st} level, or "trunk" of the Hierarchical Poly Tree, represents the final assembled composite version of the FE model. The L^{th} level, or outermost "branches" of the Tree, represents the substructures comprised of fundamental finite elements, e.g. 4-node quadrilateral elements. The intermediate levels represent the various assemblages

of the hierarchy that are used to traverse from one extreme to the other. Referring to Figure 2.2, the algorithmic steps associated with the grafting of "branch" substructures to their "root" substructure is as follows;

1. Forward Phase

   (a) Assembly of the condensed stiffness matrices and force vectors from the preceeding level,

   (b) Partitioning of the local stiffness matrix into its internal and external components. Note that this can be accomplished by simply employing the proper node numbering locally,

   (c) Forward elimination/condensation of the local stiffness matrix,

   (d) Condensation of the local force vector,

   (e) Transfer of the condensed stiffness matrix and force vector to succeeding levels, i.e. from each branch to its root processor;

2. Backward Phase

   (a) Backward transfer of the results calculated at the root substructure to its branch substructures,

   (b) Adjustment of the local internal "load" vector,

   (c) Condensation of the locally adjusted internal "load" vector,

   (d) Back substitution, i.e. calculation of the independent variables within the external periphery of the branch substructures.

In terms of Figures 2.1 and 2.2, it follows that each root-branch system must undergo the steps noted above in the Forward and Backward phases of the solution process. For multilevel trees, each branch processor is itself a root for subsequent sets of branches.

From a sequential point of view, the HPT approach to the problem is to optimize

$$C_{TS} = \sum_{l=1}^{L} \sum_{s=1}^{S(l)} \left\{ \frac{1}{2} {}_l^s N_E \left[{}_l^s N_B \left({}_l^s N_B + 1 \right) \right] + \left({}_l^s N_E \right) \left({}_l^s N_B \right) + 2 \left({}_l^s N_{E_l} \right) \left({}_l^s N_{B_l} \right) \right\} \tag{2.5}$$

where the superscript s and subscript l denote the $s^{\underline{th}}$ substructure on the $l^{\underline{th}}$ level, and

C_{TS} – total computational effort arising from solving every level and substructure in a sequential manner,

${}_l^s N_{E_I}$ – number of internal equations,

${}_l^s N_{B_I}$ – mean-half-bandwidth of the $[{}_l^s K_{II}]$ partition,

L – total number of levels,

$S(l)$ – number of substructures on the $l^{\underline{th}}$ level.

$\frac{1}{2}\,{}_l^s N_E\,[{}_l^s N_B\,({}_l^s N_B + 1)]$ – approximate number of arithmetic operations required to condense $[{}_l^s K]$

$({}_l^s N_E)\,({}_l^s N_B)$ – approximate number of arithmetic operations needed to condense $\{{}_l^s F\}$

$2\,({}_l^s N_{E_I})\,({}_l^s N_{B_I})$ – number of arithmetic operations needed to factorize $\{{}_l^s F_I\}$ and perform the subsequent back substitution step for calculating $\{{}_l^s Y_I\}$,

When operating in a multiprocessor environment the various substructures on a given level can be solved/condensed concurrently. As a result, the HPT is configured to minimize the following

$$C_{TP} = \sum_{l=1}^{L} \left\{ \frac{1}{2}\,{}_l^s N_E\,[{}_l^s N_B\,({}_l^s N_B + 1)] + ({}_l^s N_E)\,({}_l^s N_B) + 2\,({}_l^s N_{E_I})\,({}_l^s N_{B_I}) \right\}_{max} \tag{2.6}$$

where the subscript max denotes the substructure on the $l^{\underline{th}}$ level requiring the maximum computational effort. By employing the definition of the mean-half-bandwidth, Equations (2.5) and (2.6) can be recast as

$$C_{TS} = \sum_{l=1}^{L} \sum_{s=1}^{S(l)} \left[\frac{1}{2} {}^{s}_{l}\beta \left(\frac{{}^{s}_{l}\beta}{{}^{s}_{l}\alpha} + 3 \right) + 2 {}^{s}_{l}\beta_I \right]$$

$$\sim \sum_{l=1}^{L} \sum_{s=1}^{S(l)} \left[\frac{1}{2} \frac{({}^{s}_{l}\beta)^2}{{}^{s}_{l}\alpha} \right] \qquad (2.7)$$

and

$$C_{TP} = \sum_{l=1}^{L} \left[\frac{1}{2} {}^{s}_{l}\beta \left(\frac{{}^{s}_{l}\beta}{{}^{s}_{l}\alpha} + 3 \right) + 2 {}^{s}_{l}\beta_I \right]_{max}$$

$$\sim \sum_{l=1}^{L} \left[\frac{1}{2} \frac{({}^{s}_{l}\beta)^2}{{}^{s}_{l}\alpha} \right]_{max} \qquad (2.8)$$

The optimal multilevel decomposition of a finite element discretization into a hierarchy of substructures can be derived by employing various functions for β and α[5,6]. A number of these functions for two-dimensional substructuring "primitives" have been generated and are compiled in the Appendix. Note that these functions are isoparametric in nature. For example, the FE models shown in Figures 2.3 and 2.4, discounting the appropriate boundary conditions and assuming they share the same number of DOF per node, yield the same β and α. The "mechanics" of obtaining an optimal HPT will be shown more rigorously in the next chapter.

In terms of the r-extension, the initial HPT description of the model would be "fixed". This is because the r-extension method does not alter the fundamental "connectivities" of the various finite elements. In other words, even though the values stored in $[K]$ will change, β and α remain constant. On the other hand, the HPT must be able to adapt "on-the-fly" when employing the h-, p-, and/or hp-extensions. In this context, the following chapter will illustrate the flexibility of the HPT to self adapt to the demands of these type of refinements.

3 SEQUENTIAL HPT FORMULATION

Ideally, the initial global FE discretization would be decomposed into a hierarchy of substructures that have been configured according to the HPT optimality criteria. Thereafter, as mesh refinement is initiated, the HPT

would sprout multilevel root-branch systems that are optimized in the local sense. This is illustrated in Figure 3.1. However, regardless of whether a global HPT has been constructed or not, the onset of localized mesh refinement is best handled by an HPT approach. Therefore, the forthcoming development will illustrate how localized multilevel HPT's should be constructed for the sequential solution of regions that have been dynamically refined by way of h-, p-, and hp-extensions. In addition, to better convey the advantages of using this type of solution strategy, we will restrict the discussion to isoparametrically square regions comprised of four node quadrilateral elements.

Upon completion of the aforementioned, we will conclude the chapter with a comparison of the computational efforts associated with the various mesh refinement techniques. As will be seen, the differences in the number of arithmetic operations required to statically condense out the internal variables of the different methods of enhanced mesh discretization can be substantial.

3.1 SEQUENTIAL HPT FOR h-EXTENSIONS

When performing a model refinement by the h-extension method, the discretization within an element/region is increased by using smaller elements (see Figure 3.2). Typically, the computational steps associated with integrating the mesh refinement into the global formulation is as follows;

1. Static condensation is employed to remove the internal DOF,
2. The proper interpolation constraints are implemented so as to maintain element to element compatibility yielding the final representation of the refined zone in terms of the original DOF about the periphery,
3. The new element/region stiffness is assembled into the global formulation.

This sequence of steps is depicted in Figure 3.3. Now, assuming the numbering scheme shown in Figure A.1, β_h and α_h have the following form,

$$\beta_h = \frac{\rho}{2}\left[6\,\rho\,(n_1)^3 - (11\,\rho - 1)\,(n_1)^2 + 6\,\rho\,n_1\right]$$

$$\sim 3\,(\rho)^2\,(n_1)^3 \qquad (3.1)$$

$$\alpha_h = \rho\,(n_1)^2 \qquad (3.2)$$

where

ρ – node density, i.e. DOF per node,
n_1 – nodal dimension of the problem, i.e. the number of nodes along an edge.

Note that in obtaining (3.1) and (3.2) from the substructure primitive in Figure A.1, we have set the nodal dimensions m and n equal to each other. Thus, the computational effort associated with statically condensing out the internal variables is

$$C_h \sim \frac{9}{2}\,(\rho)^3\,(n_1)^4 \qquad (3.3)$$

To solve this problem with a two level, sequential HPT, the solution process is as follows;

1. The region of refinement is decomposed into an arbitrary number of equivalent substructures as depicted in Figure 3.4,

2. The individual substructures are condensed into their external DOF,

3. The condensed substructures are subsequently assembled to yield the composite structure shown in Figure 3.5,

4. The composite assembly is then statically condensed into the external DOF defining the periphery of the refined element/region,

5. The proper interpolation constraints are implemented so as to maintain element to element compatibility, yielding the final representation of the refined zone in terms of the original DOF.

6. The new element/region stiffness is assembled into the global formulation (see Figure 3.6).

Notice that the only difference between the HPT approach and the standard technique is in the way the internal variables are condensed out of the problem.

Recalling Equation (2.7), the computational effort of condensing out the internal variables of the refined element/region by a two level HPT in a sequential manner can be written as

$$ {}_hC_{TS} = \frac{1}{2}\frac{({}^1_1\beta)^2}{{}^1_1\alpha} + \sum_{s=1}^{(K_2)^2}\left[\frac{1}{2}\frac{({}^s_2\beta)^2}{{}^s_2\alpha}\right] \tag{3.4} $$

where $(K_2)^2$ is the total number of $2^{\underline{nd}}$ level substructures. Furthermore, the functionalities of ${}_1\alpha$ and ${}_1\beta$ for $K_2 \geq 3$ are, from Figure A.2,

$$ {}^1_1\alpha = \{2\left[(K_2)^2 + K_2\right] n_2 - \left[3(K_2)^2 + 2K_2 - 1\right]\}\,\rho \tag{3.5} $$

$$ \begin{aligned} {}^1_1\beta = \frac{\rho}{2}\{&[14(K_2)^3 + 7(K_2)^2 - 5K_2]\rho\,(n_2)^2 \\ &- \{[37(K_2)^3 + 13(K_2)^2 - 18K_2]\rho - [2(K_2)^2 + 2K_2]\}n_2 \\ &+ \{[24(K_2)^3 + 5(K_2)^2 - 14K_2 + 1]\rho - [3(K_2)^2 + 2K_2 - 1]\}\} \end{aligned} \tag{3.6} $$

The computational effort associated with the second level substructures is approximated by using the functions obtained from Figure A.1, i.e.

$$ \left.\begin{aligned} {}^s_2\alpha &= \rho\,(n_2)^2 \\ {}^s_2\beta &= \tfrac{\rho}{2}[6\rho\,(n_2)^3 - (11\rho - 1)(n_2)^2 + 6\rho\,n_2] \end{aligned}\right\};\ s \in [1,(K_2)^2] \tag{3.7} $$

where

$$ n_2 = \frac{n_1 + K_2 - 1}{K_2} \tag{3.8} $$

Due to the construct of the decomposition, all of the second level substructures will exhibit the same computational effort. Equation (3.4) can then be rewritten as

$$ {}_hC_{TS} \sim \frac{1}{2}\frac{({}_1\beta)^2}{{}_1\alpha} + \frac{(K_2)^2}{2}\frac{({}_2\beta)^2}{{}_2\alpha} \tag{3.9} $$

Note that, for convenience, we have discarded the use of the superscript s. Assuming $n_1 \gg K_2$, the problem dimension of the second level substructures is, from (3.8),

$$n_2 \sim \frac{n_1}{K_2} \tag{3.10}$$

Incorporating this assumption and (3.10) into (3.5) - (3.7) yields

$${}_1\alpha \sim 2(K_2 + 1)\rho n_1 \tag{3.11}$$

$${}_1\beta \sim \frac{7\rho^2}{2}(2K_2 + 1)(n_1)^2 \tag{3.12}$$

$${}_2\alpha \sim \frac{\rho(n_1)^2}{(K_2)^2} \tag{3.13}$$

$${}_2\beta \sim \frac{3\rho^2(n_1)^3}{(K_2)^3} \tag{3.14}$$

Employing (3.11) - (3.14) in (3.9) and requiring that

$$\frac{d({}_hC_{TS})}{d(K_2)} = 0 \tag{3.15}$$

yields

$$K_2 \sim \left[\frac{36}{49} n_1\right]^{1/3} \quad ; K_2 \geq 3 \tag{3.16}$$

Thus, the proper number of second level substructures needed to minimize ${}_hC_{TS}$ is given by (3.16). The computational speedup afforded by this approach is illustrated by forming the ratio

$$R_{h/TS} = \frac{C_h}{{}_hC_{TS}} \tag{3.17}$$

Recall that ${}_hC_{TS}$ and C_h are the approximate number of arithmetic operations required to condense out the internal variables of the refined element/region with and without substructuring respectively. Utilizing the result of (3.16), it can be shown that

$$R_{h/TS} = \frac{C_h}{{}_hC_{TS}} \sim 0.31(n_1)^{2/3} \quad ; K_2 \geq 3 \tag{3.18}$$

Focussing on the special case of $K_2 = 2$, the functions for ${}_1\alpha$ and ${}_1\beta$ are (from Figure A.3),

$$\begin{aligned} {}_1\alpha &= (6\,n_1 - 9)\,\rho \\ &\sim 6\,\rho\,n_1 \end{aligned} \tag{3.19}$$

$$\begin{aligned} {}_1\beta &= \frac{\rho}{4}[65\rho(n_1)^2 - (182\rho - 12)\,n_1 + (123\rho - 18)] \\ &\sim \frac{65}{4}\,\rho^2\,(n_1)^2 \end{aligned} \tag{3.20}$$

and

$$\begin{aligned} n_2 &= \frac{n_1 + 1}{2} \\ &\sim \frac{n_1}{2} \end{aligned} \tag{3.21}$$

It is easily verified that the speedup attainable from this decomposition of the problem is

$$R_{h/TS} = \frac{C_h}{{}_hC_{TS}} \sim \frac{864\,n_1}{(4225 + 216\,n_1)} \;;\; K_2 = 2 \tag{3.22}$$

Moreover,

$$\lim_{n_1 \to \infty} R_{h/TS} = 4 \;;\; K_2 = 2 \tag{3.23}$$

Figure 3.7 graphically illustrates the potential speedups that can be obtained for a sequentially solved two level HPT with $K_2 = 2, 3$, and 4 for h-extended mesh refinements where $n_1 \leq 60$.

The preceeding development has been based upon the assumption that the problem size, n_1, is large. At this juncture it is appropriate to ask the question, 'How large must n_1 be before the benefits of the HPT are realized?' To address this question, a number of numerical experiments were performed. The speedups obtained from this empirical study are depicted in Figures 3.8 - 3.11 and tabulated in Tables (3.1) - (3.3). It can be clearly seen that as the problem size increases, the speedups are as predicted by the foregoing development. In addition, the following observations can be made:

1. Small problems are dominated by the computational overhead. However, for larger problems, this effect becomes negligible;

2. Decomposing the problem into four substructures ($K_2 = 2$) is more advantageous than nine ($K_2 = 3$) for small n_1 because less overhead is accrued;

3. As the DOF per node increases, i.e. ρ, the results are improved for small values of n_1. This occurs because, for a given problem size n_1, the overhead becomes less influential as a result of the actual computational effort increasing by an order of $\mathcal{O}(\rho^3)$.

4. For larger problems, the second level substructures themselves become large enough to warrant another level of substructuring.

It was also found, as evidenced by Tables (3.1) - (3.3), that the system of equations resulting from this type of mesh refinement can be condensed/ solved more efficiently by utilizing a multilevel HPT with $K_l = 2, l \in [2, L]$ where the number of levels that should be employed is governed by

$$\rho(n_{L-1})^2 \geq 350 \tag{3.24}$$

Equation (3.24) arises from the fact that whenever $\rho\,(n_L)^2$ is greater than 350, another level of substructuring is justified so long as $K_{L+1} = 2$. Thus, to determine the approximate number of levels that should be used in terms of ρ and n_1, n_{L-1} can be cast as

$$\begin{aligned} n_{L-1} &= \frac{n_1 + \prod_{l=2}^{L-1} K_l - 1}{\prod_{l=2}^{L-1} K_l} \\ &\sim \frac{n_1}{\prod_{l=2}^{L-1} K_l} \end{aligned} \tag{3.25}$$

Requiring that $K_l = 2$, $l \in [2, L]$,

$$n_{L-1} \sim \frac{n_1}{(2)^{(L-2)}} \tag{3.26}$$

Substituting (3.26) into (3.24) yields

$$L \leq \frac{3}{4}\left\{\ln[\rho\,(n_1)^2] - 3\right\} \tag{3.27}$$

Before proceeding, it must be emphasized that the limiting number of levels is based upon empirical data. That is, the computational overhead incurred varies from one machine to the next and is the dominant factor in determining the smallest problem size, $\rho(n_1)^2$, that can benefit from this type of solution strategy.

To estimate the performance of an L level HPT solved sequentially, wherein $K_l = 2$, $l \in [2, L]$, we can write

$$ {}_hC_{TS} = \frac{1}{2}\left[\frac{({}_1\beta)^2}{{}_1\alpha} + 4\,\frac{({}_2\beta)^2}{{}_2\alpha} + 16\,\frac{({}_3\beta)^2}{{}_3\alpha} + \cdots + (2)^{2(L-1)}\,\frac{({}_L\beta)^2}{{}_L\alpha}\right] \tag{3.28} $$

where, for $l \in [1, L-1]$,

$$ {}_l\alpha = (6n_l - 9)\,\rho \sim 6\,\rho\,n_l \tag{3.29} $$

$$ {}_l\beta = \frac{\rho}{4}[65\,\rho\,(n_l)^2 - (182\rho - 12)\,n_l + (123\rho - 18)] \sim \frac{65}{4}\,\rho^2\,(n_l)^2 \tag{3.30} $$

and

$$ n_l \sim \frac{n_1}{(2)^{(l-1)}} \quad ; l \in [2, L] \tag{3.31} $$

$$ {}_L\alpha = \rho\,(n_L)^2 \tag{3.32} $$

$$ {}_L\beta = \frac{\rho}{2}\,[6\,\rho\,(n_L)^3 - (11\rho - 1)\,(n_L)^2 + 6\,\rho n_L] \sim 3\,\rho^2\,(n_L)^3 \tag{3.33} $$

Using (3.29) - (3.33), we can recast (3.28) as

$$ {}_hC_{TS} = \frac{4225}{192}\,\rho^3\,(n_1)^3\left[\sum_{l=1}^{L-1}\frac{1}{(2)^{(l-1)}}\right] + \frac{9}{2}\,\rho^3\frac{(n_1)^4}{(2)^{2(L-1)}} $$

$$ = \frac{4225}{192}\,\rho^3\,(n_1)^3\left[2 - \frac{1}{(2)^{(L-2)}}\right] + \frac{9}{2}\,\rho^3\,\frac{(n_1)^4}{(2)^{2(L-1)}} \tag{3.34} $$

As before, the potential speedup is estimated by forming the ratio

$$R_{h/TS} = \frac{C_h}{{}_hC_{TS}}$$

$$= \frac{432\,(2)^{2(L-1)}\,n_1}{\{4225\,(2)^{(L-1)}\,[(2)^{(L-1)} - 1] + 432\,n_1\}} \qquad (3.35)$$

Equations (3.27) and (3.35) can be used in conjunction to determine the appropriate number of levels and the concomitant speedup that can be expected for a given h-extended refinement defined by ρ and n_1. It is also interesting to note that, for a given fixed value of L, the speedup for large values of n_1 is

$$\lim_{n_1 \to \infty} R_{h/TS} = (2)^{2(L-1)} \;\; ; K_l = 2, \;\; l \in [2, L] \qquad (3.36)$$

Another important feature inherent to the HPT solution strategy is the reduction of the memory needed to store the stiffness matrix elements. However, even though the total storage requirements are reduced, the substructures on the $2^{\underline{nd}}$ through $L^{\underline{th}}$ levels must store the partition of the stiffness matrices containing the connectivities between the internal and external D.O.F., i.e. $[K_{IE}]$, separately before it is altered by the condensation process. This is because the internal load vector, $\{F_I\}$, of a branch substructure must be adjusted by way of

$$\{F_I\}_{adj.} = \{F_I\} - [K_{IE}]\{Y_E\} \qquad (3.37)$$

to reflect the influence of the external displacements, $\{Y_E\}$, calculated by the root.

In this context, the number of stiffness matrix elements that must be stored for an $l^{\underline{th}}$ level substructure can be written as

$$M_1 = {}_1\beta \qquad (3.38)$$

$$M_l = {}_l\beta + {}_l\beta_{IE} \;\; ; l \in [2.L] \qquad (3.39)$$

where

M_l – total number of stiffness matrix elements that must be stored

${}_l\beta$ – number of stiffness matrix elements stored in $[K]$ via a skylined technique

${}_l\beta_{IE}$ – number of stiffness matrix elements in the $[K_{IE}]$ partition of $[K]$

When using the optimal HPT configuration of $K_l = 2$, $l \in [2, L]$, the functions for ${}_l\beta$ are defined by (3.30) and (3.33) while

$$
\begin{aligned}
{}_l\beta_{IE} &= \frac{\rho^2}{4}[26(n_l)^2 - 84n_l + 50] \\
&\sim \frac{13}{2}\rho^2(n_l)^2 \quad ; l \in [1,(L-1)] \qquad (3.40)
\end{aligned}
$$

$$
\begin{aligned}
{}_L\beta_{IE} &= \rho^2[2(n_L)^3 - 8(n_L)^2 + 10n_L - 4] \\
&\sim 2\rho^2(n_L)^3 \qquad (3.41)
\end{aligned}
$$

Using (3.31), (3.40), and (3.41); (3.38) and (3.39) can be written as

$$M_1 \sim \frac{65}{4}\rho^2(n_1)^2 \qquad (3.42)$$

$$M_l \sim \frac{91}{4}\rho^2\frac{(n_1)^2}{(2)^{2(l-1)}} \quad ; l \in [2,(L-1)] \qquad (3.43)$$

$$M_L \sim 5\rho^2\frac{(n_1)^3}{(2)^{3(L-1)}} \qquad (3.44)$$

Then, for an L level HPT, the total storage needed for all the various stiffness matrices can be cast as

$$M_{TS} = M_1 + (4)\,M_2 + (16)\,M_3 + \cdots + (2)^{2(L-1)}\,M_L \qquad (3.45)$$

From (3.42) - (3.44),

$$ {}_hM_{TS} = \frac{65}{4}\rho^2(n_1)^2 + \frac{91}{4}\rho^2(L-2)(n_1)^2 + 5\rho^2\frac{(n_1)^3}{(2)^{(L-1)}} \qquad (3.46) $$

Thus, the overall reduction in memory requirements afforded from the use of an HPT solution strategy can be seen by ratioing ${}_hM_{TS}$ with β_h, namely

$$ \begin{aligned} {}_hM_{TS/h} &= \frac{{}_hM_{TS}}{\beta_h} \\ &\sim \frac{\left[\frac{65}{4}\rho^2(n_1)^2 + \frac{91}{4}\rho^2(L-2)(n_1)^2 + \frac{5\rho^2(n_1)^3}{(2)^{(L-1)}}\right]}{3\rho^2(n_1)^3} \\ &\sim \frac{65}{12n_1} + \frac{91(L-2)}{12n_1} + \frac{5}{3(2)^{(L-1)}} \; ; \quad K_l = 2, \quad l \in [2, L] \end{aligned} \qquad (3.47) $$

Fixing the number of levels and letting n_1 become large, we see that

$$ \lim_{n_1 \to \infty} {}_hM_{TS/h} = \frac{5}{3}\frac{1}{(2)^{(L-1)}} \; ; K_l = 2, \; l \in [2, L] \qquad (3.48) $$

Figure 3.12 graphically illustrates the reduced number of stiffness matrix elements that must be stored when employing an HPT for h-extended mesh refinements. As can be seen, an actual savings of memory can not be realized until $n_1 \geq 22$. The fact that the memory requirements are increased for smaller h-extensions is primarily attributable to the dual storage of the $[K_{IE}]$ partition of the stiffness matrix.

The foregoing development has shown that the use of a multilevel substructuring approach, i.e. the HPT, for solving locally refined mesh discretizations by h-extension has certain distinct advantages. First, the internal variables are statically condensed out of the refined element/region much more efficiently and, secondly, substantial reductions in the overall memory requirements are achieved. In addition, although less quantifiable, is the local "error trapping" provided by this technique. This is due to the reduction of a given degree of freedoms skyline height resulting from the substructured decomposition of the problem. In other words, the direct

influence of a finite element approximation error in a particular stiffness matrix element is confined to the degrees of freedom within the reduced skyline. The HPT also provides a logical and efficient means of "telescoping" an h-extended mesh refinement into a physical anomaly or singularity. This is done by grafting another localized HPT onto the previous one as shown in Figure 3.13. In the section that follows, we will develop the advantages and quantifiable trends of using the HPT in a sequential manner for p- and hp-extended mesh refinements.

3.2 SEQUENTIAL HPT FOR p- AND hp-EXTENSIONS

In the previous section it was shown that a locally h-extended mesh discretization can be solved more efficiently by using the Hierarchical Poly Tree solution strategy. In this section we will show that the HPT is applicable to p- and hp-extensions as well. As before, we will restrict the discussion to isoparametrically square regions. Although the development will be similar to the one presented for the h-extension technique, the approach will be quite different. This is due to

1. The discretization within a given element is enhanced by using higher ordered polynomial shape functions; and

2. The number of elements refined in this manner is independent of the order of the polynomial chosen to improve their accuracy.

In this context, to maintain compatibility with the equations used in the previous section, the independent variables defining this type of mesh refinement will be κ and η. The interpretation of these variables is as follows:

$(\kappa)^2$ – is the total number of elements refined by p-extension; and

η – quantifies the order of the polynomial used within the elements themselves by way of the number of nodes along an edge.

With regards to the order of the polynomial used, the prevalent literature typically uses complete $p^{\underline{th}}$ order polynomials for triangular elements

as defined by Pascal's triangle[9]. Therefore, to be consistent, we will define the $p^{\underline{th}}$ order polynomial of a 4-node quadrilateral element as the conjunction of two complete $p^{\underline{th}}$ order triangular elements as shown in Figure 3.14. Furthermore, the variables η and p are related by

$$\eta = p + 1 \tag{3.49}$$

As was pointed out by Katz, Peano, and Rossow[10], the internal variables arising from the increase in p are condensed out at the element level. Katz et al. also showed that the order of the polynomial can be increased by enforcing constraints on the higher order derivatives of the shape functions as opposed to the use of extra spatial nodes. Regardless of which method is used, if only C^0 continuity is required, the number of arithmetic operations needed to statically condense out the internal DOF for an element refined in this manner is

$$\begin{aligned} C'_p = \frac{1}{24}\{&\rho^3[(\eta)^6 + 9(\eta)^5 + 30(\eta)^4 - 391(\eta)^3 + 921(\eta)^2 - 822\eta + 256] \\ &+ \rho^2[3(\eta)^4 + 30(\eta)^3 - 195(\eta)^2 + 312\eta - 132] \\ &- \rho[10(\eta)^2 - 40\eta + 40]\} \end{aligned} \tag{3.50}$$

It can also be shown that the number of stiffness matrix elements for the previously defined $p^{\underline{th}}$ order 4-node quadrilateral element is

$$\beta_p = \frac{\rho}{4}\{[(\eta)^3 + 6(\eta)^2 - 11\eta + 6]\rho + 2\eta\}\eta \tag{3.51}$$

Equations (3.50) and (3.51) were derived for the numbering scheme given in Figure A.4. Moreover, after condensation, the resulting form of the refined element is shown in Figure 3.15. In addition, the order of the complete polynomials, i.e. p, are usually restricted to values of 8 or less because of the numerical error incurred while calculating the appropriate coefficients[11]. It then follows that

$$\eta \leq 9 \tag{3.52}$$

Referring to Figure 3.16, an isoparametrically square region comprised of $(\kappa)^2$ elements refined by p-extension is integrated into the original, global FE model by performing the following algorithmic steps:

1. The internal DOF generated by the p-extension refinement are condensed out at the element level;

2. The $(\kappa)^2$ refined elements are subsequently assembled;

3. The internal DOF associated with this assemblage are condensed into the DOF about the periphery of the refined region;

4. The appropriate interpolation constraints that will maintain element to element compatibility are applied; and

5. The refined region is assembled into the global FE model via the original DOF about the periphery.

The number of arithmetic operations needed to perform steps 1) and 3) can be approximated as

$$C_{hp} = \frac{1}{2}\frac{(\beta_{hp})^2}{\alpha_{hp}} + (\kappa)^2 C_p \tag{3.53}$$

where, from (3.5) and (3.6),

$$\alpha_{hp} = \{2[(\kappa)^2 + \kappa]\eta - [3(\kappa)^2 + 2\kappa - 1]\}\rho \tag{3.54}$$

$$\begin{aligned}\beta_{hp} = \frac{\rho}{2}\{&[14(\kappa)^3 + 7(\kappa)^2 - 5\kappa]\rho(\eta)^2 \\ &- \{[37(\kappa)^3 + 13(\kappa)^2 - 18\kappa]\rho - [2(\kappa)^2 + 2\kappa]\}\eta \\ &+ \{[24(\kappa)^3 + 5(\kappa)^2 - 14\kappa + 1]\rho - [3(\kappa)^2 + 2\kappa - 1]\}\}\end{aligned} \tag{3.55}$$

Assuming $\kappa \gg \eta \geq 3$ yields

$$\alpha_{hp} \sim \rho(\kappa)^2(2\eta - 3) \tag{3.56}$$

$$\beta_{hp} \sim \frac{\rho^2}{2}(\kappa)^3[14(\eta)^2 - 37\eta + 24] \tag{3.57}$$

Substituting (3.56) and (3.57) into (3.53) gives the approximate computational effort entailed in performing a large scale p-extended mesh refinement, that is

$$C_{hp} \sim \frac{\rho^3}{8}\kappa^4 \frac{(14\eta^2 - 37\eta + 24)^2}{(2\eta - 3)} + \kappa^2 C_p \tag{3.58}$$

Note that, due to the way it was formulated, (3.58) is also applicable to hp-extensions. This is because it is written in terms of the number of elements and the order of the polynomial enhancement within them, i.e. $(\kappa)^2$ and $\eta = (p + 1)$ respectively.

In the previous section it was shown that an optimal sequential solution of an h-extended mesh refinement can be obtained from a HPT configured such that $K_l = 2$, $l \in [2, L]$. The forthcoming discussion will show that the same is true when addressing p- and hp-extended mesh discretizations. To set the stage for the generalized L level case, we will illustrate the development of a simple two level HPT where $K_2 = 2$. Recalling (3.9) and (3.50), the computational effort for a two level HPT can be written as

$$\begin{aligned} {}_{hp}C_{TS} &= \frac{1}{2}\frac{({}_1\beta)^2}{{}_1\alpha} + \frac{(K_2)^2}{2}\frac{({}_2\beta)^2}{{}_2\alpha} + (\kappa)^2 C_p \\ &\sim \frac{1}{2}\frac{({}_1\beta)^2}{{}_1\alpha} + 2\frac{({}_2\beta)^2}{{}_2\alpha} + (\kappa)^2 C_p \end{aligned} \tag{3.59}$$

Rewriting (3.19) and (3.20),

$${}_1\alpha \sim 6\rho n_1 \tag{3.60}$$

$${}_1\beta \sim \frac{65}{4}\rho^2 (n_1)^2 \tag{3.61}$$

where, from Figure 3.17 and appealing to Equation (3.8),

$$\begin{aligned} n_1 &= \kappa(\eta - 1) + 1 \\ &\sim \kappa(\eta - 1) \end{aligned} \tag{3.62}$$

Substituting (3.62) into (3.60) and (3.61) gives

$$ {}_1\alpha \sim 6\rho\kappa(\eta - 1) \tag{3.63} $$

$$ {}_1\beta \sim \frac{65}{4}\rho^2(\kappa)^2(\eta - 1)^2 \tag{3.64} $$

Once again referring to Figure 3.17 and assuming that $(\kappa/2) \geq 3$, we see that the operative functions for ${}_2\alpha$ and ${}_2\beta$ are equivalent to (3.5) and (3.6), namely

$$ {}_2\alpha = \left\{2\left[\left(\frac{\kappa}{2}\right)^2 + \frac{\kappa}{2}\right]\eta - \left[3\left(\frac{\kappa}{2}\right)^2 + 2\left(\frac{\kappa}{2}\right) - 1\right]\right\}\rho \tag{3.65} $$

$$ \begin{aligned} {}_2\beta = \frac{\rho}{2}\Bigg\{&\left[14\left(\frac{\kappa}{2}\right)^3 + 7\left(\frac{\kappa}{2}\right)^2 - 5\left(\frac{\kappa}{2}\right)\right]\rho(\eta)^2 \\ &- \left\{\left[37\left(\frac{\kappa}{2}\right)^3 + 13\left(\frac{\kappa}{2}\right)^2 - 18\left(\frac{\kappa}{2}\right)\right]\rho - \left[2\left(\frac{\kappa}{2}\right)^2 + 2\left(\frac{\kappa}{2}\right)\right]\right\}\eta \\ &+ \left\{\left[24\left(\frac{\kappa}{2}\right)^3 + 5\left(\frac{\kappa}{2}\right)^2 - 14\left(\frac{\kappa}{2}\right) + 1\right]\rho - \left[3\left(\frac{\kappa}{2}\right)^2 + 2\left(\frac{\kappa}{2}\right) - 1\right]\right\}\Bigg\} \end{aligned} \tag{3.66} $$

Applying the assumption that κ is large and much greater than η yields

$$ {}_2\alpha \sim \rho\left(\frac{\kappa}{2}\right)^2(2\eta - 3) \tag{3.67} $$

$$ {}_2\beta \sim \frac{\rho^2}{2}\left(\frac{\kappa}{2}\right)^3[14(\eta)^2 - 37\eta + 24] \tag{3.68} $$

Utilizing (3.63),(3.64),(3.67), and (3.68); (3.59) can be recast as

$$ {}_{hp}C_{TS} = \frac{4225}{192}\rho^3(\kappa)^3(\eta - 1)^3 + \frac{\rho^3}{32}(\kappa)^4\frac{[14(\eta)^2 - 37\eta + 24]^2}{(2\eta - 3)} + (\kappa)^2 C_p \tag{3.69} $$

The resulting speedup obtained from such a decomposition is

$$R_{hp/TS} = \frac{C_{hp}}{{}_{hp}C_{TS}}$$

$$\sim \frac{\left\{\frac{\rho^3}{8}(\kappa)^4\frac{[14(\eta)^2-37\eta+24]^2}{(2\eta-3)} + (\kappa)^2 C_p\right\}}{\left\{\frac{4225}{192}\rho^3(\kappa)^3(\eta-1)^3 + \frac{\rho^3}{32}(\kappa)^4\frac{[14(\eta)^2-37\eta+24]^2}{(2\eta-3)} + (\kappa)^2 C_p\right\}} \tag{3.70}$$

Speedups obtained for isoparametrically square regions refined by a p-extension and decomposed into a two level HPT is shown in Figures 3.18a and 3.18b. Upon inspection of these graphs the following observations can be made

1. The speedup afforded by an HPT increases as the problem size increases; and

2. For a given κ, speedup may actually improve for a higher order of p.

Lastly, before expounding the generalized sequential L level HPT, it can be shown that, for a fixed value of η, the asymptotic speedup is

$$\lim_{\kappa\to\infty} R_{hp/TS} = 4 \; ; K_2 = 2 \tag{3.71}$$

Note that (3.71) correlates with the result of (3.23).

Moving on to the L level HPT, the recursion formula of (3.31) can be written in terms of κ and η, namely

$$n_l \sim \frac{n_1}{(2)^{(l-1)}}$$

$$\sim \frac{\kappa(\eta-1)}{(2)^{(l-1)}} \tag{3.72}$$

Substituting (3.72) into (3.29) and (3.30) yields, for $l \in [1,(L-1)]$,

$$ {}_{l}\alpha \sim 6\rho\frac{\kappa(\eta-1)}{(2)^{(l-1)}} \tag{3.73} $$

$$ {}_{l}\beta \sim \frac{65}{4}\rho^2\frac{(\kappa)^2(\eta-1)^2}{(2)^{2(l-1)}} \tag{3.74} $$

$$ \frac{1}{2}\frac{({}_{l}\beta)^2}{{}_{l}\alpha} \sim \frac{4225}{192}\rho^3\frac{(\kappa)^3(\eta-1)^3}{(2)^{3(l-1)}} \tag{3.75} $$

Finally, appealing to (3.67) and (3.68),

$$ {}_{L}\alpha \sim \rho\frac{(\kappa)^2}{(2)^{2(L-1)}}(2\eta-3) \tag{3.76} $$

$$ {}_{L}\beta \sim \frac{\rho^2}{2}\frac{(\kappa)^3}{(2)^{3(L-1)}}[14(\eta)^2-37\eta+24] \tag{3.77} $$

$$ \frac{1}{2}\frac{({}_{L}\beta)^2}{{}_{L}\alpha} \sim \frac{\rho^3}{8}\frac{(\kappa)^4}{(2)^{4(L-1)}}\frac{[14(\eta)^2-37\eta+24]^2}{(2\eta-3)} \tag{3.78} $$

Employing (3.50), (3.75), and (3.78); the number of arithmetic operations required to solve an L level HPT sequentially in terms of κ and η is

$$ \begin{aligned} {}_{hp}C_{TS} &= \sum_{l=1}^{L}\frac{(2)^{2(l-1)}}{2}\frac{({}_{l}\beta)^2}{{}_{l}\alpha} + (\kappa)^2 C_p \\ &\sim \frac{4225}{192}\rho^3(\kappa)^3(\eta-1)^3\left[\sum_{l=1}^{L-1}\frac{1}{(2)^{(l-1)}}\right] \\ &\quad + \frac{\rho^3}{8}\frac{(\kappa)^4}{(2)^{2(L-1)}}\frac{[14(\eta)^2-37\eta+24]^2}{(2\eta-3)} + (\kappa)^2 C_p \\ &= \frac{4225}{192}\rho^3(\kappa)^3(\eta-1)^3\left[2-\frac{1}{(2)^{(L-2)}}\right] \\ &\quad + \frac{\rho^3}{8}\frac{(\kappa)^4}{(2)^{2(L-1)}}\frac{[14(\eta)^2-37\eta+24]^2}{(2\eta-3)} + (\kappa)^2 C_p \end{aligned} \tag{3.79} $$

Ratioing (3.58) with (3.79) gives the approximate speedup that can be expected from employing an L level HPT. Moreover, fixing L and η, the speedup of a localized p- or hp-extended mesh refinement where κ is large can be shown to be

$$\lim_{\kappa \to \infty} R_{hp/TS} = (2)^{2(L-1)} \; ; \; K_l = 2, \; l \in [2, L] \qquad (3.80)$$

The results of (3.36) and (3.80) indicate that asymptotically large mesh discretization refinements that are isoparametrically square, regardless of whether its an h-, p-, or hp-extension; can be solved with similar computational improvements when using multilevel HPT decompositions.

Before the relative storage requirements of the standard and HPT solutions can be quantified for p- and/or hp-extensions, some subtle issues pertaining to this type of mesh refinement must be addressed. For example, if the internal DOF within the p-extended elements themselves are to be calculated, then each individual element stiffness matrix must be saved for the back substitution phase of the solution. In addition, the elements unaltered $[K_{IE}]$ partition must be saved separately so that the internal load vector can be formulated as per (3.37). Furthermore, even if the internal unknowns are not desired, one may still wish to save the individual element stiffness matrices anyway. This arises from the fact that the recalculation of the entire element stiffness matrix can be avoided if an increase in the order of p is needed as the solution progressed from one iteration to the next. As was shown by Katz et. al.[10], special nodal variables can be created so that the updated element stiffness matrix can be constructed by simply appending the rows and columns of the new DOF to the initial matrix.

Keeping the aforementioned issues in mind, the number of stiffness matrix elements stored when not using the HPT can be written as

$$M_{hp} = \beta_{hp} + \kappa^2(\beta_p +_p \beta_{IE}) \qquad (3.81)$$

where

$$\beta_p = \text{Equation (3.51)}$$

$$\beta_{hp} = \text{Equation (3.55)}$$

$$_p\beta_{IE} = \frac{\rho}{2}[(6\eta^3 - 11\eta^2 + 5\eta + 2)\rho + 4(\eta - 1)] \qquad (3.82)$$

Note that $_p\beta_{IE}$ accounts for the dual storage of the connectivity partition of the stiffness matrix for the p-extended elements. If the internal DOF of the individual elements are not calculated, this term can be discarded from (3.81). Moreover, if the element stiffness matrix is regenerated from scratch when needed, β_p can be thrown out as well.

For the HPT, the functions given by (3.42) and (3.43) for an h-extension are applicable to p- and hp-extensions as well. Writing them in terms of κ and η yields

$$M_1 \sim \frac{65}{4}\rho^2\kappa^2(\eta - 1)^2 \qquad (3.83)$$

$$M_l \sim \frac{91}{4}\frac{\rho^2\kappa^2(\eta - 1)^2}{(2)^{2(l-1)}} \quad ; l \in [2,(L-1)] \qquad (3.84)$$

The number of stiffness matrix elements stored for the $L^{\underline{th}}$ level can be represented as

$$M_L = (_L\beta + _L\beta_{IE}) + \frac{\kappa^2}{(2)^{2(L-1)}}(\beta_p + _p\beta_{IE}) \qquad (3.85)$$

Assuming that the $L^{\underline{th}}$ level substructure is comprised of at least nine p-extended elements, i.e.

$$\frac{\kappa}{(2)^{(L-1)}} \geq 3 \qquad (3.86)$$

then ${}_L\beta$ is defined by (3.77) and

$$
\begin{aligned}
{}_L\beta_{IE} &= \frac{\rho^2}{2}\left[\left(\frac{8\kappa^3}{(2)^{3(L-1)}} - \frac{\kappa^2}{(2)^{2(L-1)}} - \frac{3\kappa}{(2)^{(L-1)}} - 2\right)\eta^2\right. \\
&\quad - \left(\frac{20\kappa^3}{(2)^{3(L-1)}} - \frac{\kappa^2}{(2)^{2(L-1)}} - \frac{7\kappa}{(2)^{(L-1)}} - 6\right)\eta \\
&\quad \left. + \left(\frac{12\kappa^3}{(2)^{3(L-1)}} - \frac{2\kappa^2}{(2)^{2(L-1)}} - \frac{2\kappa}{(2)^{(L-1)}} - 4\right)\right] \\
&\sim \frac{\rho^2}{2}\frac{\kappa^3}{(2)^{3(L-1)}}(8\eta^2 - 20\eta + 12) \qquad (3.87)
\end{aligned}
$$

From (3.45) and (3.83) - (3.87), the total number of stiffness matrix elements stored when using the HPT is

$$
\begin{aligned}
{}_{hp}M_{TS} &\sim \frac{\rho^2\kappa^2(\eta-1)^2}{4}(91L - 117) + \frac{\rho^2}{2}\frac{\kappa^3}{(2)^{(L-1)}}(22\eta^2 - 57\eta + 36) \\
&\quad + \kappa^2(\beta_p +_p \beta_{IE}) \qquad (3.88)
\end{aligned}
$$

Using (3.81) and (3.88), the relative reduction in the number of stiffness matrix elements that must be stored by the HPT, for a large number of p-extended elements, is

$$
\lim_{\kappa\to\infty} {}_{hp}M_{TS/hp} \sim \frac{1}{(2)^{(L-1)}}\frac{(22\eta^2 - 57\eta + 36)}{(14\eta^2 - 37\eta + 24)} \; ; K_l = 2, \; l \in [2, L] \qquad (3.89)
$$

Setting the HPT solution strategy aside for a moment, it is worthwhile to note the significant difference in the number of computational operations required to condense out the internal variables of an element refined by h-extension as opposed to p-extension. That is, assuming the same number of DOF per node and $n_1 = \eta$, the number of arithmetic operations required to statically condense out the internal DOF of an h- extended element is, from (3.3), on the order of $\mathcal{O}[\rho^3(n_1)^4]$ and, from (3.42), on the order of $\mathcal{O}[\rho^3(\eta)^6]$ for the p-extended version.

This observation is significant in that the p-extension, for a fixed number of DOF, is generally accepted to be the more accurate of the two methods[11,12]. However, as was just shown, this occurs at the cost of being much more computationally intensive. In the next section we will discuss more rigorously the implications of the relative computational efforts associated with h-, p-, and hp-extended elements.

3.3 COMPARISON OF h-, p-, AND hp-EXTENSIONS

To date, a great deal of effort has been expended to determine the relative accuracy of the h-, p-, and hp-extension techniques[3,11-13]. The criteria generally used to make this comparison is the amount of error incurred for a given number of DOF. But, with the same number of DOF, the static condensation process for the various methods of localized mesh refinement can have substantially different computational costs. These differences will be quantified on a relative basis in this section. Obviously, there are other aspects of the solution, beyond the actual condensation process, that can affect the overall computational effort. However, our objective here is to simply point out that the actual CPU time required to obtain a given degree of accuracy might be a more relevent basis of comparison.

To begin the discussion, we will compare the methods of h- and p-extensions as applied to a single element. Before proceeding, however, certain aspects of the comparison must be clarified. For example, when referring to a p-extended element, it should be understood that we are addressing the conjunction of two complete $p^{\underline{th}}$ order triangular elements as described in the previous section. In addition, since it is not clear as to whether the internal DOF of a p-extended element will be calculated or not, we will restrict the discussion to the computational effort associated with condensing the stiffness matrices and load vectors. In other words, the back substitution phase of the solution will not be considered. With this in mind, the number of arithmetic operations required to statically condense out the internal variables of an h-extended element is, from (3.3),

$$C_h \sim \frac{9}{2}\rho^3(n_1)^4 \qquad (3.90)$$

For a p-extended element, the number of computations required to perform the same operation is, from (3.50),

$$C_p \sim \frac{\rho^3}{24}\eta^5(\eta+9) \tag{3.91}$$

Since we are addressing the refinement of a single element, n_1 and η are equivalent, see Figures 3.2 and 3.14. That is, assuming the same number of DOF per node (ρ), n_1 and η define the same total number of DOF. Therefore, from (3.90) and (3.91),

$$\frac{C_h}{C_p} \sim \frac{108}{n_1(n_1+9)} \tag{3.92}$$

As can be seen from (3.92), the amount of computational effort for a single element refined by p-extension exceeds that of an h-extended one when $n_1 = \eta$. To further convey the difference in the cost of the two approaches, a plot of the actual and theoretical ratio of C_h/C_p as a function of n_1 is shown in Figure 3.19. This difference can be accounted for by the realization that a p-extended element gives rise to a stiffness matrix that is very nearly full while an h-extended one is essentially banded. The significance of this observation can be better appreciated when you consider that, from (2.4),

$$C_h \sim \frac{1}{2}\frac{(\beta_h)^2}{\alpha_h} \tag{3.93}$$

$$C_p \sim \frac{1}{2}\frac{(\beta_p)^2}{\alpha_p} \tag{3.94}$$

where

$$\alpha_h = \alpha_p = \rho(n_1)^2 = \rho\eta^2 \tag{3.95}$$

$$\beta_h = \text{Equation (3.1)}$$

$$\beta_p = \text{Equation (3.51)}$$

It then follows that

$$\frac{C_h}{C_p} \sim \left(\frac{\beta_h}{\beta_p}\right)^2 \tag{3.96}$$

Thus, any relative reduction in the number of stiffness matrix elements afforded by an h-extension is amplified by the nonlinearity of (3.96). As an example, from Figure 3.20,

$$\frac{\beta_h}{\beta_p} \sim \frac{1}{2} \quad ; n_1 = \eta = 15 \tag{3.97}$$

Substituting (3.97) into (3.96) gives

$$\frac{C_h}{C_p} \sim \frac{1}{4} \quad ; n_1 = \eta = 15 \tag{3.98}$$

Note that the result of (3.98) correlates with the value shown in Figure 3.19 for the prescribed level of refinement.

Proceeding onto a comparison of the h- and hp-extension techniques, the number of arithmetic operations needed to condense an hp-extended element/region can be approximated as, from (3.58) and (3.91),

$$C_{hp} \sim \frac{\rho^3}{8}\kappa^4\frac{(14\eta^2 - 37\eta + 24)^2}{(2\eta - 3)} + \frac{\rho^3}{24}\kappa^2\eta^5(\eta + 9) \tag{3.99}$$

Recalling that n_1 can be written in terms of κ and η by (3.62), C_h can be recast as

$$C_h \sim \frac{9}{2}\rho^3[\kappa(\eta - 1) + 1]^4 \tag{3.100}$$

Using (3.99) and (3.100), Figure 3.21 shows the resulting ratio of C_h/C_{hp} for various values of κ and η. As can be seen, C_h is greater than C_{hp} for all the combinations of κ and η presented. Since it was shown earlier that C_p is greater than C_h, this result may seem to be contradictory. The explanation for the apparent discrepency is as follows. When performing an hp-extension, the individual p-extended elements condense out their internal DOF before they are assembled into the composite version of the refined mesh. Recall that this sequence of algorithmic steps is illustrated in Figure 3.16. As a result, an hp-extension constructed in such a manner is inherently a two level HPT wherein the $2^{\underline{nd}}$ level substructures are the p-extended elements. Then, just like the HPT, if the size of the $2^{\underline{nd}}$ level substructures (p-extended elements) becomes too large with respect to their subsequent assemblage defined by K_2 (κ), the computational effort will

become suboptimal. Thus, if the order of the polynomials within the p-extended elements were allowed to become sufficiently large, C_h would be less than C_{hp}. This behavior, for the smaller values of κ, can be seen from Figure 3.21. To obtain this relation, however, would require orders of p that are impractical and typically not used.

To get a better feel for the computational efficiency obtained from performing an hp-extension in the aforementioned manner, we will decompose the h-extension into the same number of substructures as p-extended elements, i.e. $K_2 = \kappa$ and $n_2 = \eta$. Consequently, the only difference between the two approaches is in the effort of condensing out the internal DOF of the substructures on the 2^{nd} level. Thus, in terms of κ and η,

$$ {}_hC_{TS} \sim \frac{\rho^3}{8}\kappa^4\frac{(14\eta^2 - 37\eta + 24)^2}{(2\eta - 3)} + \frac{9}{2}\rho^3\kappa^2(\eta - 1)^4 \qquad (3.101) $$

Ratioing (3.101) with (3.99) gives

$$ \frac{{}_hC_{TS}}{C_{hp}} \leq 1 \quad ;\kappa \geq \eta \geq 3 \qquad (3.102) $$

More quantitatively, Figure 3.22 shows ${}_hC_{TS}/C_{hp}$ over the same range of values for κ and η used in Figure 3.21. Comparing these two plots with each other, we see that the use of localized condensation methods yields a more favorable comparison of computational efficiency for an h-extension. Moreover, as the number of substructures (κ) increases, the ratio ${}_hC_{TS}/C_{hp}$ approaches unity. This occurs because the assemblage of the substructures is dominating the solution. Since the composite version of the mesh refinement is the same for both methods, it stands to reason that,from (3.99) and (3.101),

$$ \lim_{\kappa \to \infty} \frac{{}_hC_{TS}}{C_{hp}} = 1 \qquad (3.103) $$

Based upon the foregoing, it is apparent that the use of substructuring techniques can significantly impact the relative computational costs. In this context, we will compare the h- and hp-extension techniques when they have been hierarchically substructured into their respective optimal HPT's. In terms of the parameters κ and η, the minimum number of arithmetic operations for an h-extension is, from (3.34),

$$ {}_hC_{TS} = \frac{4225}{192}\rho^3\kappa^3(\eta - 1)^3\left[2 - \frac{1}{(2)^{(L-2)}}\right] + \frac{9}{2}\frac{\rho^3\kappa^4(\eta - 1)^4}{(2)^{2(L-1)}} \qquad (3.104) $$

where

$$\tfrac{3}{4}\{\ln[\rho(\kappa\eta-\kappa+1)^2]-3\}-1 \;\leq\; {}_hL_{opt.} \;\leq\; \tfrac{3}{4}\{\ln[\rho(\kappa\eta-\kappa+1)^2]-3\} \tag{3.105}$$

The computational effort for an hp-extension disseminated into its optimal HPT is, from (3.79),

$$\begin{aligned} {}_{hp}C'_{TS} \;\sim\; & \frac{4225}{192}\rho^3(\kappa)^3(\eta-1)^3\left[2-\frac{1}{(2)^{(L-2)}}\right] \\ & + \frac{\rho^3}{8}\frac{(\kappa)^4}{(2)^{2(L-1)}}\frac{[14(\eta)^2-37\eta+24]^2}{(2\eta-3)} \;+\; (\kappa)^2\,C'_p \end{aligned} \tag{3.106}$$

Since the functionality of (3.106) was derived with the assumption that the $L^{\underline{th}}$ level substructures are comprised of at least nine p-extended elements, the maximum number of levels that can be employed is obtained from the constraint

$$\left[\frac{\kappa}{(2)^{(L-1)}}\right]^2 \geq 3^2 \tag{3.107}$$

Solving for L yields

$$\frac{\ln(\kappa/3)}{\ln(2)} \leq {}_{hp}L_{opt.} \leq \frac{\ln(2\kappa/3)}{\ln(2)} \tag{3.108}$$

Referring to the plot of ${}_hC'_{TS}/{}_{hp}C'_{TS}$, i.e. Figure 3.23, it is very interesting to note the strong influence that an additional level of substructuring can have on the efficiency of an hp-extension. This influence is manifested through the racheting behavior clearly seen at the appropriate values of κ, i.e. $\kappa = 6, 12$, and 24. The effect of additional levels does diminish, however, as κ gets larger. This is evidenced by the progressively smaller "step" sizes at the transition values of κ. Furthermore, the relative efficiency of an h-extension improves as η increases. This is due, in part, to the fact that an h-extension can add more levels to handle the increase in the total

number of DOF arising from larger values of η. Conversely, the number of levels for an hp-extension is strictly controlled by the number of p-extended elements (κ).

Lastly, regardless of the value of η, Figure 3.23 shows that ${}_hC_{TS}/{}_{hp}C_{TS}$ is approaching unity as κ becomes large. This is indicative of the fact that the lower levels of the HPT are dominating the solution time. Since the total DOF have been constrained to be the same for both methods, the h- and hp-extensions have equivalent computational efforts for the lower levels of the HPT. From this we can conclude, if the total DOF is sufficiently large, that the multilevel HPT provides a computational efficiency which is invariant to the type of fundamental finite element used in the model.

Another facet of the solution process that could significantly impact the total CPU time required to perform an h-, p-, or hp-extension is the generation and assembly of the additional elements. In general, both the number and type of element should be considered. In many instances, however, only one element actually needs to be created. Occasions such as this arise when the refined mesh is comprised of elements with the same geometry and aspect ratios. Although our comparison of the various methods has not taken this part of the solution into account, it has sufficed to show that the computational effort not only varies from one technique to the other, but is strongly dependent upon the solution strategy as well. In this context, we have satisfied our objective. That is, the actual CPU time required to obtain a given degree of accuracy should at least be included in any real comparison of the various mesh refinement techniques.

4 PARALLEL HPT FORMULATION

In the previous chapter it was assumed that the multilevel substructural decomposition of the locally refined mesh discretization had to be condensed/solved one substructure at a time. Since multiprocessor computers are becoming more commonplace, the forthcoming development will be based upon the premise that the substructures occurring on any particular level can be condensed/solved concurrently. As was shown by Padovan and Gute [6], this approach to the solution of FE type numerical models can yield significant computational improvements. In many instances the speedup will be even greater than the number of processors used, i.e.

superlinear.

Superlinearity is a measure of the processor usage efficiency. More specifically, superlinearity is the ratio of the effective speedup with the number of processors used, i.e.

$$S_g = \frac{R_{g/TP}}{\Phi} \tag{4.1}$$

where

S_g – superlinearity
$R_{g/TP}$ – effective speedup
Φ – number of processors used.

It is the opinion of the authors that this approach to measuring the efficiency of processor usage is more appropriate than other conventional measures. This arises from the fact that, as will be seen, the number of substructures/processors required to obtain the optimal effective speedup is problem dependent. From this it follows that arbitrarily disseminating an FE model into the same number of substructures as there are available processors will typically lead to suboptimal results. In addition, the effective speedup will be determined by comparing the effective computational effort of the parallel HPT solution with that of the standard sequential solution. This is done for two reasons. First, it is a measure of speedup that the general FE user community can identify with as a result of their almost exclusive use of single processor, sequential type computers. Second, using the standard sequential solution as a reference forms the basis from which the efficiency of all parallel solution algorithms can be compared.

Overall, there are many factors that will affect the actual speedups obtained on a multiprocessor computer. These include:

1. The communication/data bus structure of the processor network;
2. The degree of sophistication of the resident compiler;
3. The amount of globally shared memory and the concomitant access efficiency;
4. The speed of the individual processors themselves, etc.

In this context, the objective here will be to simply illustrate the potential advantages of using the HPT solution strategy to integrate a local mesh refinement into the initial FE model when a parallel network of processors can be exploited. This will include trends associated with the following:

1. Effective speedup;
2. Approximation of superlinearity; and
3. Reduction of memory requirements.

Moreover, we will show how the superlinearity of the solution can be improved, without drastically degrading the potential speedup, by implementing a technique called Top-Down, Partial Sequentialism (TDPS)[6]. Furthermore, based on our earlier comments, it will be assumed that:

1. The time required to transfer data from one level of the hierarchy to the next is negligible;
2. All of the processors share the same computational capacity as the sequential reference; and
3. Each processor has enough local in-core memory to store the data of its assigned substructure.

4.1 PARALLEL HPT FOR *h*-EXTENSIONS

In Chapter 2 it was shown that the effective computational effort associated with the parallel solution of a hierarchically substructured FE model can be approximated as

$$C'_{TP} \sim \sum_{l=1}^{L} \left[\frac{1}{2} \frac{({}^{s}_{l}\beta)^2}{{}^{s}_{l}\alpha} \right]_{max} \tag{4.2}$$

or, more succinctly,

$$C'_{TP} \sim \sum_{l=1}^{L} (C'_l)_{max} \tag{4.3}$$

where

$$(C_l)_{max} = \left[\frac{1}{2}\frac{({}^s_l\beta)^2}{{}^s_l\alpha}\right]_{max} \tag{4.4}$$

Since the substructures occurring on any particular level are constructed so that they are computationally equivalent, we will, for convenience, dispense with the max subscript. Recall that this type of substructure construction is possible because we have restricted the discussion to isoparametrically square regions of mesh refinement.

For a two level HPT, wherein the substructures on the $2^{\underline{nd}}$ level can be condensed/solved simultaneously, (4.2) can be written as

$$C_{TP} = \frac{1}{2}\frac{({}_1\beta)^2}{{}_1\alpha} + \frac{1}{2}\frac{({}_2\beta)^2}{{}_2\alpha} \tag{4.5}$$

Addressing the solution of a local h-extension, the appropriate functionalities for ${}_l\alpha$ and ${}_l\beta$, $l \in [1,2]$, are defined by (3.11) - (3.14). Note that (3.11) and (3.12) are only valid for $K_2 \geq 3$. Substituting these functions into (4.5) yields

$${}_hC_{TP} \sim \frac{49}{16}\rho^3\frac{(2K_2+1)^2}{(K_2+1)}(n_1)^3 + \frac{9}{2}\rho^3\frac{(n_1)^4}{(K_2)^4} \tag{4.6}$$

Thus, the approximate number of processors/substructures that should be used on the $2^{\underline{nd}}$ level is determined by satisfying

$$\frac{d({}_hC_{TP})}{d(K_2)} = 0 \tag{4.7}$$

Solving (4.7) yields

$$K_2 \sim \left(\frac{72}{49}n_1\right)^{1/5} \quad ; n_1 \geq 165 \tag{4.8}$$

$$K_2 = 3 \quad ; n_1 < 165 \tag{4.9}$$

Under most circumstances the value of n_1 can be expected to be less than 165. Consequently, for our present purposes, we will assume that $K_2 = 3$. Substituting this into (4.6) gives

$${}_hC_{TP} \sim \frac{2401}{64}\rho^3(n_1)^3 + \frac{1}{18}\rho^3(n_1)^4 \tag{4.10}$$

Ratioing C_h, as given by (3.3), with (4.10) gives the effective speedup of this decomposition. More specifically,

$$\begin{aligned} R_{h/TP} &= \frac{C_h}{{}_hC_{TP}} \\ &\sim \frac{\left[\frac{9}{2}\rho^3(n_1)^4\right]}{\left[\frac{2401}{64}\rho^3(n_1)^3 + \frac{1}{18}\rho^3(n_1)^4\right]} \\ &= \frac{2592\,n_1}{(21609 + 32\,n_1)} \end{aligned} \tag{4.11}$$

To set the stage for the measure of superlinearity, we can write the number of processors of an L level HPT, in general, as

$$\Phi = 1 + \sum_{l=2}^{L} \left(\prod_{i=2}^{l} K_i \right)^2 \tag{4.12}$$

For $L = 2$ and $K_2 = 3$, (4.12) gives

$$\begin{aligned} \Phi &= 1 + (K_2)^2 \\ &= 10 \end{aligned} \tag{4.13}$$

Ratioing (4.11) with (4.13) gives the approximate superlinearity that can be expected, i.e.

$$\begin{aligned} S_h &= \frac{R_{h/TP}}{\Phi} \\ &\sim \frac{2592n_1}{10(21609 + 32n_1)} \end{aligned} \tag{4.14}$$

As was stated in the introduction of this chapter, the actual performance of an HPT on a given multiprocessor computer is dependent upon several factors. However, by using the empirical data obtained for the sequential HPT, we can predict the actual speedup and superlinearity of a parallel HPT within a reasonable percentage of error. Thus, Figures 4.1 through 4.4 graphically show the correlation of the theoretical speedups and

superlinearities with the "actual" for various problem sizes when $L = 2$ and $K_2 = 3$.

For the case of $K_2 = 2$, the operative functions of ${}_1\alpha$ and ${}_1\beta$ are defined by (3.19) and (3.20) respectively. Incorporating these into (4.5) yields

$$ {}_hC_{TP} \sim \frac{4225}{192}\rho^3(n_1)^3 + \frac{9}{32}\rho^3(n_1)^4 \qquad (4.15) $$

Once again, forming the appropriate ratio with (3.3), i.e. C_h, and (4.15) gives the speedup for a two level HPT with $K_2 = 2$. Namely,

$$ R_{h/TP} = \frac{864n_1}{(4225 + 54n_1)} \qquad (4.16) $$

From (4.12), the number of processors required to obtain the speedup defined by (4.16) is

$$ \Phi = 5 \quad ; L = 2, K_2 = 2 \qquad (4.17) $$

It then follows that the superlinearity of this particular decomposition is

$$ S_h \sim \frac{864n_1}{5(4225 + 54n_1)} \qquad (4.18) $$

Figures 4.5 - 4.8 show how the speedup and superlinearity vary with problem size for $L = 2$ and $K_2 = 2$. Moreover, from Tables (4.1) and (4.2), the following observations can be made:

1. The use of four processors ($K_2 = 2$) on the second level provides faster speedups than nine ($K_2 = 3$) for problems where $n_1 \leq 55$;

2. The magnitude of superlinearity is greater when using five processors instead of ten for problem sizes in the range of interest; and

3. Regardless of whether four or nine processors are used on the $2^{\underline{nd}}$ level, both the speedup and superlinearity improve as the problem size increases.

Furthermore, from Table (4.3), a three level HPT with $K_2 = K_3 = 2$ provides a better effective speedup than a two level tree for modest values of n_1. In this context we can conclude, as with the sequential HPT, that the optimal effective speedup for a locally h-extended element/region can

be obtained with a parallel HPT configuration wherein $K_l = 2$, $l \in [2, L]$, i.e.

$$R_{h/TP} \sim \frac{6048(2)^{4(L-1)}n_1}{\{4225(2)^{4(L-1)}[8-(2)^{-3(L-2)}]+6048n_1\}} \qquad (4.19)$$

This conclusion amends the result of $K_l = 3$, $l \in [2, L]$, that was given in [6] where the special case of $K_l = 2$ was not investigated. Fixing the number of levels and allowing n_1 to become large, the asymptotic speedup can readily be seen to be

$$\lim_{n_1 \to \infty} R_{h/TP} = (2)^{4(L-1)} \;\; ; K_l = 2,\; l \in [2, L] \qquad (4.20)$$

Moreover, the number of levels that should be employed to attain the optimal effective speedup can be approximated by satisfying

$$\rho(n_{L-1})^2 \geq 160 \qquad (4.21)$$

As with Equation (3.24), (4.21) was determined from empirical data and is likely to vary from one machine to the next. Using the recursion formula of (3.31), (4.21) can be recast in terms of n_1, namely

$$\frac{\rho(n_1)^2}{(2)^{2(L-2)}} \geq 160 \qquad (4.22)$$

Solving for L yields

$$L \leq \frac{18}{25}\{\ln[\rho(n_1)^2] - 2.3\} \qquad (4.23)$$

Although (4.23) will give the number of levels that will yield the optimal effective speedup, the use of this many levels can severely degrade the superlinearity of the network. This phenomena is clearly indicated in Table (4.3). To maintain the speedups afforded by the addition of more levels, while improving the superlinearity, we can employ the technique of Top - Down, Partial Sequentialism[6]. TDPS, as the name implies, performs the condensation of the higher levels (Top) of the HPT in a sequential manner while solving the lower levels in parallel. The fact that TDPS will not significantly degrade the effective speedup can be seen by forming the ratios

$$C_{l/h} = \frac{C_l}{C_h} \;\; ; \; l \in [1, L] \qquad (4.24)$$

Letting $K_l = 2, l \in [2, L]$, where $L \geq 2$, Equation (4.24) can be written as

$$C_{l/h} \sim \frac{4225}{864} \frac{1}{(2)^{3(l-1)} n_1} \; ; l \in [1,(L-1)] \tag{4.25}$$

$$C_{L/h} \sim \frac{1}{(2)^{4(L-1)}} \tag{4.26}$$

From (4.25) we see that

$$C_{1/h} = 8C_{2/h} = 64C_{3/h} = \cdots = (2)^{3(L-2)} C_{(L-1)/h} \tag{4.27}$$

In addition, $C_{(L-1)/h}$ is greater than $C_{L/h}$ so long as

$$n_1 < \frac{4225}{216}(2)^L \tag{4.28}$$

Equation (4.27) clearly shows that the computational effort of the higher levels is much less than the lower ones. It then follows that solving the higher levels sequentially will not impinge upon the overall effective speedup. Consequently, the superlinearity of the solution can be improved because substantially fewer processors are used to obtain essentially the same speedup. This is evidenced by Table (4.4) where the 2^{nd} level processors were also used to solve the 3^{rd} level substructures sequentially. In general, the computational effort associated with the use of the TDPS technique can be written as

$$\begin{aligned} {}_hC_{TDPS} \sim & \sum_{l=1}^{L-\mathcal{L}-1} \left[\frac{4225}{192} \frac{\rho^3 (n_1)^3}{(2)^{3(l-1)}} \right] \\ & + \sum_{l=L-\mathcal{L}}^{L-1} \left[\frac{4225}{192} \frac{(2)^{2(l-L+\mathcal{L})}}{(2)^{3(l-1)}} \rho^3 (n_1)^3 \right] \\ & + \frac{9}{2} \frac{(2)^{2\mathcal{L}}}{(2)^{4(L-1)}} \rho^3 (n_1)^4 \\ = & \frac{4225}{192} \rho^3 (n_1)^3 \left\{ \frac{1}{7} [8 - (2)^{-3(L-\mathcal{L}-2)}] + (2)^{(4+2\mathcal{L}-3L)} [(2)^{\mathcal{L}} - 1] \right\} \\ & + \frac{9}{2} \rho^3 (n_1)^4 (2)^{(4+2\mathcal{L}-4L)} \end{aligned} \tag{4.29}$$

where

$\mathcal{L}$ – the number of higher levels solved sequentially, $0 \leq \mathcal{L} \leq (L-1)$

L – the total number of levels, including thosed solved sequentially

The total number of processors necessitated by the use of TDPS can be written in terms of L and $\mathcal{L}$ also, namely

$$\begin{aligned} \Phi &= 1 + \sum_{l=2}^{L-\mathcal{L}} (2)^{2(l-1)} \\ &= \frac{1}{3}[(2)^{2(L-\mathcal{L})} - 1] \end{aligned} \tag{4.30}$$

The reduction in the total number of stiffness matrix elements that must be stored for the parallel HPT is the same as that given by (3.47) and (3.48). However, for networks that do not have globally shared memory capabilities, it is worthwhile to note the reduced memory requirements on a per processor basis. Utilizing (3.38) and (3.39) with the proper functionalities for ${}_l\beta$ and ${}_l\beta_{lE}$, the fractional memory needs of the processors on different levels are given by

$$M_{l/h} = \frac{M_l}{\beta_h} \quad ; \, l \in [1, L] \tag{4.31}$$

Figure 4.9 depicts (4.31) for $l \in [1, (L-1)]$ in terms of the parameter n_1. As can be seen, the use of the HPT solution strategy on a parallel network of processors for h-extended mesh discretizations can significantly reduce the memory demands placed on a given processor. This is especially true for processors employed on the higher levels of the Tree.

If the TDPS technique is used, care must be taken to ensure that the available memory resources of the processor performing the computations of the higher levels does not become saturated. In terms of the variables L, $\mathcal{L}$, and n_1; the number of stiffness matrix elements that must be stored by an $(L-\mathcal{L})^{\underline{th}}$ level processor when using TDPS is

$$\begin{aligned} {}_{TDPS}M_{(L-\mathcal{L})} &= M_{L-\mathcal{L}} + (4)M_{(L-\mathcal{L}+1)} + \cdots + (2)^{2\mathcal{L}} M_L \\ &\sim \rho^2 (n_1)^2 [91\mathcal{L}(2)^{2(\mathcal{L}-L)} + 5n_1(2)^{(3+2\mathcal{L}-3L)}] \end{aligned} \tag{4.32}$$

It has been shown that the dissemination of an h-extended element/region into a multilevel hierarchy of substructures can provide substantial, even superlinear, speed enhancements. To improve the solution characteristics even further, more standard parallel solution schemes could be implemented in conjunction with the HPT. For example, since the processors on levels that are not currently "active" are essentially "idle", they could be used to perform the condensation of the "active" substructures via the "Parallel Active Equation Solver" developed by Farhat and Wilson [14]. Moreover, the assemblages of the lower levels of the hierarchy would lend themselves very well to such solution techniques because of their relatively large skyline heights.

4.2 PARALLEL HPT FOR p- AND hp-EXTENSIONS

The advantages of using an HPT solution strategy on a parallel network of processors were presented in the previous section for h-extended mesh refinements. In this section we will reformulate the parallel HPT in terms of the parameters used to describe p- and hp-extensions, that is κ and η. Once again, we will be concerned with demonstrating the effective speedup, superlinearity, and memory requirements provided by the HPT approach.

To begin the discussion, recall that the approximate computational effort of solving this type of mesh refinement without a HPT was given by (3.58). Now, assuming that the $2^{\underline{nd}}$ level substructures are solved concurrently, the effective number of arithmetic operations incurred by a two level HPT can be approximated as

$$ {}_{hp}C_{TP} \sim \frac{1}{2}\frac{({}_1\beta)^2}{{}_1\alpha} + \frac{1}{2}\frac{({}_2\beta)^2}{{}_2\alpha} + \left(\frac{\kappa}{K_2}\right)^2 C_p \qquad (4.33) $$

Note that (4.33) also employs the assumption that the internal DOF of the individual p-extended elements are condensed out simultaneously by the processors they were assigned to on the $2^{\underline{nd}}$ level. Letting $K_2 = 2$. Equations (3.75) and (3.78) can be used to rewrite (4.33) as

$$ {}_{hp}C_{TP} \sim \frac{4225}{192}\rho^3\kappa^3(\eta-1)^3 + \frac{\rho^3\kappa^4}{128}\frac{(14\eta^2-37\eta+24)^2}{(2\eta-3)} + \frac{\kappa^2}{4}C_p \qquad (4.34) $$

Ratioing (3.58) with (4.34) gives the effective speedup obtained for a parallel, two level HPT where $\kappa \gg \eta \geq 3$ and $K_2 = 2$. Figure 4.10 shows the potential speedup as a function of κ and η for this type of HPT decomposition. It can also be shown that as the number of p-extended elements becomes large that the speedup is bounded by

$$\lim_{\kappa \to \infty} R_{hp/TP} = 16 \quad ; K_2 = 2 \tag{4.35}$$

This result is consistent with the two level asymptotic speedup given by (4.20) for large h-extended elements/regions.

The number of processors used for a two level Tree when $K_2 = 2$ is five, see (4.17). The superlinearity of the HPT for this set of circumstances is shown in Figure 4.11. As can be seen from Figures 4.10 and 4.11, both speedup and superlinearity improve as the number of p-extended elements is increased.

For a general L level HPT wherein $K_l = 2$, $l \in [2, L]$, the approximate computational effort as a function of κ and η is

$$\begin{aligned} {}_{hp}C_{TP} \quad \sim \quad & \frac{4225}{1344}\rho^3\kappa^3(\eta-1)^3[8-(2)^{-3(L-2)}] \\ & + \frac{\rho^3\kappa^4}{8(2)^{4(L-1)}}\frac{(14\eta^2-37\eta+24)^2}{(2\eta-3)} \\ & + \frac{\kappa^2}{(2)^{2(L-1)}}C_p \end{aligned} \tag{4.36}$$

The speedup potential for a given number of levels, L, from (3.58) and (4.36), is

$$\lim_{\kappa \to \infty} R_{hp/TP} = (2)^{4(L-1)} \quad ; K_l = 2,\ l \in [2, L] \tag{4.37}$$

The bounded speedups given by (3.36), (3.80), (4.20), and (4.37) are indicative of the fact that the substructure assemblages occurring on levels less than the $L^{\underline{th}}$ have relative computational efforts that are inversely proportional to the problem size. In other words, for h-extensions,

$$\frac{C_l}{C_h} = \mathcal{O}\left(\frac{1}{n_1}\right) \quad ; l \in [1,(L-1)] \tag{4.38}$$

or, for p- and hp-extensions,

$$\frac{C_l}{C_{hp}} = \mathcal{O}\left[\frac{1}{\kappa(\eta-1)}\right] \quad ; l \in [1.(L-1)] \tag{4.39}$$

In this context, the speedups obtained from a multilevel HPT is only limited by the number of levels which can be used to scale down the FE model from its global form to that of a much smaller L^{th} level substructure.

From the perspective of superlinearity, the number of processors for an L level HPT with $K_l = 2$, $l \in [2, L]$, can be written as

$$\begin{aligned} \Phi &= \frac{1}{3}[(2)^{2L} - 1] \\ &\sim \frac{(2)^{2L}}{3} \end{aligned} \tag{4.40}$$

Using (4.37) and (4.40), the superlinearity for an asymptotically large mesh refinement with a fixed value of L is

$$\begin{aligned} \lim_{\kappa \to \infty} S_{hp} &= \frac{\lim_{\kappa \to \infty} R_{hp/TP}}{\Phi} \\ &\sim \frac{3}{16}(2)^{2L} \end{aligned} \tag{4.41}$$

For more typical values of κ, the most efficient use of the processor network would be obtained by using the technique of TDPS. In general,

$$\begin{aligned} {}_{hp}C'_{TDPS} \sim\ & \frac{4225}{192}\rho^3\kappa^3(\eta-1)^3\left\{\frac{1}{7}[8-(2)^{-3(L-\mathcal{L}-2)}] + (2)^{(4+2\mathcal{L}-3L)}[(2)^{\mathcal{L}}-1]\right\} \\ & + (2)^{2\mathcal{L}}\left[\frac{\rho^3\kappa^4}{8(2)^{4(L-1)}}\frac{(14\eta^2-37\eta+24)^2}{(2\eta-3)} + \frac{\kappa^2}{(2)^{2(L-1)}}C'_p\right] \end{aligned} \tag{4.42}$$

Figures 4.12 and 4.13 show the speedup and superlinearity for various values of κ and η when $L = 3$ and $\mathcal{L} = 1$. Comparing these with Figures 4.10 and 4.11, one can see that the technique of TDPS not only improves superlinearity, but, for larger numbers of p-extended elements, can enhance the overall effective speedup as well.

The total number of stiffness matrix elements stored by the HPT for this type of mesh refinement is given by (3.88) and (3.89). On a per processor/substructure basis, the relative storage requirements can be posed in the same manner as (4.31). The operative functions for p- and hp-extensions

are given by (3.81) and (3.83) - (3.85). More specifically,

$$M_{1/hp} \sim \frac{\left[\frac{65}{4}\rho^2\kappa^2(\eta-1)^2\right]}{[\beta_{hp}+\kappa^2(\beta_p+_p\beta_{IE})]} \tag{4.43}$$

$$M_{l/hp} \sim \frac{\left[\frac{91}{4}\frac{\rho^2\kappa^2(\eta-1)^2}{(2)^{2(l-1)}}\right]}{[\beta_{hp}+\kappa^2(\beta_p+_p\beta_{IE})]} \quad ; l \in [2,(L-1)] \tag{4.44}$$

$$M_{L/hp} = \frac{\left[(_L\beta+_L\beta_{IE})+\frac{\kappa^2}{(2)^{2(L-1)}}(\beta_p+_p\beta_{IE})\right]}{[\beta_{hp}+\kappa^2(\beta_p+_p\beta_{IE})]} \tag{4.45}$$

Comparing (4.43) with (4.44) we see that the storage requirements on the first level exceeds those of the processors on the $2^{\underline{nd}}$ through $(L-1)^{\underline{th}}$ levels, that is

$$M_{1/hp} > M_{2/hp} > \cdots > M_{l/hp} > \cdots > M_{(L-1)/hp} \tag{4.46}$$

In this context, the first level forms an upper bound of the memory requirements for the processors used on the first $(L-1)$ levels. This upper bound is shown in terms of κ and η in Figure 4.14. As can be seen from Figure 4.14, the relative reduction in the number of stiffness matrix elements that must be stored on a per processor basis improves as the number of p-extended elements increases. Upon inspection of (4.45) it is apparent that the relative storage requirements for the $L^{\underline{th}}$ level processors depends on L, κ, and η. However, since the relative storage requirements for the $L^{\underline{th}}$ level decreases as L increases, the limiting fractional storage requirements for this level occurs when $L=2$. Thus, from (4.45), it can be shown that

$$M_{L/hp} \leq \frac{1}{4} \quad ; \eta \leq 10 \tag{4.47}$$

To conclude our discussion on the use of the HPT for p- and hp-extended mesh discretizations in a parallel computing environment, the total number of stiffness matrix elements that must be stored by the $(L-\mathcal{L})^{\underline{th}}$ level processor when using TDPS is

$$\begin{aligned} {}_{TDPS}M_{(L-\mathcal{L})} &= \sum_{l=L-\mathcal{L}}^{L-1}(2)^{(l-L+\mathcal{L})}M_l + (2)^{2\mathcal{L}}M_L \\ &\sim 91\rho^2\kappa^2(\eta-1)^2\mathcal{L}(2)^{2(\mathcal{L}-L)} + (2)^{2\mathcal{L}}M_L \end{aligned} \tag{4.48}$$

Equation (4.48) can be used to ensure that the processor performing the calculations of the higher levels will not have its local memory resources saturated when employing TDPS. Note that this applies only for machines that do not have "globally" shared memory facilities. Furthermore, (4.48) is similar to (4.32) in that the only differnce is in the amount of storage necessitated by the $L^{\underline{th}}$ level substructures. This arises from the fact that (4.48) can account for the storage of the stiffness matrices associated with the p-extended elements and their subsequent assemblage into the $L^{\underline{th}}$ level substructure.

5 SUMMARY

This paper has demonstrated how a multilevel substructuring technique, called the Hierarchical Poly Tree (HPT), can be used to integrate a local mesh refinement into the original finite element model more efficiently. The optimal HPT configurations for solving isoparametrically square regions of mesh refinement on single and multiple processor computers was derived. Moreover, it was also shown that the HPT inherently reduces the total number of stiffness matrix elements that must be stored. For example, an h-extension of an element/region can be solved sequentially on a single processor computer with a speedup approximated by

$$R_{h/TS} = \frac{C_h}{{}_hC_{TS}} \sim \frac{432\,(2)^{2(L-1)}\,n_1}{\{4225\,(2)^{(L-1)}\,[(2)^{(L-1)} - 1] + 432\,n_1\}} \tag{5.1}$$

As can be seen, the speedup afforded by the HPT is dependent upon the size of the mesh refinement and the number of substructuring levels used, i.e. n_1 and L respectively. However, for a given value of L, the asymptotic speedup for large n_1 is

$$\lim_{n_1 \to \infty} R_{h/TS} = (2)^{2(L-1)} \tag{5.2}$$

In addition, the fractional number of stiffness matrix elements that must be stored when using the HPT solution strategy was shown to be

$${}_hM_{TS/h} = \frac{{}_hM_{TS}}{M_h} \sim \frac{(91L - 117)}{12n_1} + \frac{5}{3(2)^{(L-1)}} \;\; ; \;\; L \geq 2 \tag{5.3}$$

The number of levels that should be employed to achieve the optimal speedup must be determined empirically. That is, the smallest substructure size that the refinement can be subdivided into, without degrading the solution time, is machine dependent. This is because computational overhead varies from one machine to the next and is the dominant factor in determining this parameter. Theoretically, the number of levels that should be used to obtain the most efficient solution would only be limited by the fact that the $L^{\underline{th}}$ level substructures must contain more than one fundamental finite element[6].

To address the solution of an hp-extended element/region, the degree of mesh refinement was defined by the variables κ and η. More specifically, $(\kappa)^2$ is the total number of p-extended elements and η quantifies the complete $p^{\underline{th}}$ order polynomial used within the elements by way of the number of "external" nodes along one edge of the periphery. Note that an hp-extension of a single element is computationally equivalent to the assemblage of $(\kappa)^2$ p-extended "global" elements of the same polynomial order. The speedup obtained by using the sequential HPT for this type of mesh refinement is given by

$$R_{hp/TS} = \frac{C_{hp}}{{}_{hp}C_{TS}} \sim \frac{\text{Equation (3.58)}}{\text{Equation (3.79)}} \tag{5.4}$$

However, if κ is large, the relation

$$n_1 \sim \kappa(\eta - 1) \tag{5.5}$$

can be used in conjunction with (5.1) to approximate the resulting speedup within a reasonable percentage of error.

As with the h-extension, the HPT solution strategy can reduce the total number of stiffness matrix elements that must be stored for an hp-extension. The actual magnitude of these savings, however, is dependent upon some subtle, but significant issues. These include:

1. If the internal DOF within the p-extended elements themselves are to be calculated, then each individual element stiffness matrix, along with its unaltered $[K_{IE}]$ partition, must be saved for the back substitution phase of the analysis; or,

2. If the special nodal variables developed by Katz et.al. [10] are used, the element stiffness matrix may be saved to avoid recalculating it from scratch whenever an increase in the order of p is needed as the solution progresses from one iteration to the next.

With this in mind, the relative number of stiffness matrix elements that must be stored when utilizing the HPT with respect to the standard hp-extension solution is

$$ {}_{hp}M_{TS/hp} = \frac{{}_{hp}M_{TS}}{M_{hp}} \sim \frac{\text{Equation (3.88)}}{\text{Equation (3.81)}} \qquad (5.6) $$

It needs to be pointed out that Equations (3.81) and (3.88) have accounted for the extra storage required to store the element stiffness matrices and corresponding $[K_{IE}]$ partitions. If these matrices do not have to be stored, the appropriate terms can simply be discarded. For a large number of p-extended elements, it was shown that

$$ \lim_{\kappa \to \infty} {}_{hp}M_{TS/hp} \sim \frac{1}{(2)^{(L-1)}} \frac{(22\eta^2 - 57\eta + 36)}{(14\eta^2 - 37\eta + 24)} \qquad (5.7) $$

The advantages of using the HPT solution strategy on a multiprocessor computer were also presented. Machines of this type provide the capability to condense/solve the substructures on any particular level concurrently. Oftentimes the effective speedup that can be obtained from exploiting this technology is even greater than the number of processors used. In particular, an h-extension solved in this manner will yield an effective speedup of

$$ R_{h/TP} = \frac{C_h}{{}_hC_{TP}} \sim \frac{6048(2)^{4(L-1)}n_1}{\{4225(2)^{4(L-1)}[8 - (2)^{-3(L-2)}] + 6048n_1\}} \qquad (5.8) $$

Recall that this measure is made relative to the standard sequential solution. This was done for the following reasons:

1. It is a measure of speedup that the general FE user community can identify with as a result of their almost exclusive use of single processor, sequential type computers; and,

2. Using the standard sequential solution as a reference forms the basis from which the efficiency of all parallel solution algorithms can be compared.

As with the sequential HPT, the potential speedup depends upon the problem size, n_1, and the number of substructuring levels, L, employed. But, fixing the value of L and letting n_1 become large yields

$$\lim_{n_1 \to \infty} R_{h/TP} = (2)^{4(L-1)} \tag{5.9}$$

Although (5.9) indicates that the use of more levels would enhance the overall speedup that could be attained, one must consider if the increased number of processors that this would require is warranted. It was in this context that a measure of the processor usage efficiency, called superlinearity, was defined. Specifically, it is the ratio of the effective speedup with the total number of processors/substructures used. This approach to quantifying the efficiency of the solution was chosen because the optimal number of processors/substructures that should be used to achieve the best speedup is problem dependent. Since the number of processors for an optimal HPT is

$$\begin{aligned} \Phi &= \frac{1}{3}[(2)^{2L} - 1] \\ &\sim \frac{(2)^{2L}}{3} \end{aligned} \tag{5.10}$$

the superlinearity for an h-extension can be approximated as

$$S_h = \frac{R_{h/TP}}{\Phi} \sim \frac{18144(2)^{(2L-4)}n_1}{\{4225(2)^{4(L-1)}[8 - (2)^{-3(L-2)}] + 6048n_1\}} \tag{5.11}$$

Thus, from (5.11), a two level HPT will provide a speedup that is greater than the number of processors used when $n_1 \geq 40$. For a three level HPT, the problem size must be such that $n_1 \geq 130$.

To achieve the speedups afforded by the addition of more substructuring levels, and still be computationally efficient for smaller sizes of n_1, a technique called Top-Down, Partial Sequentialism (TDPS)[6] can be used. TDPS takes advantage of the fact that the higher levels of the hierarchy represent a small portion of the total computational effort. Subsequently, the substructures on the higher levels of the HPT can be solved sequentially by processors assigned to the lower levels without significantly impinging the overall solution time. As an example, for an h-extension with $n_1 = 37$,

it was shown that the third level substructures of a three level HPT solved sequentially by the second level processors would yield a superlinearity greater than one while still "conserving" 81 percent of the speedup. Both the superlinearity and effective speedup obtained from using TDPS improve as the problem size increases.

For hp-extensions solved by a parallel HPT, the effective speedup that can be expected, in terms of κ and η, is

$$R_{hp/TP} = \frac{C_{hp}}{{}_{hp}C_{TP}} \sim \frac{\text{Equation (3.58)}}{\text{Equation (4.36)}} \tag{5.12}$$

Using (5.12), it can be shown that the limiting speedup for a given number of substructuring levels is

$$\lim_{\kappa \to \infty} R_{hp/TS} = (2)^{4(L-1)} \tag{5.13}$$

Note that the asymptotic speedups provided by sequential and parallel HPT's are invariant with respect to the mode of refinement used. In other words, regardless of whether the refined mesh discretization is a large scale h- or hp-extension, the relative speedups will be the same. In fact, it was shown that the actual solution times will be essentially the same for a given number of DOF. This follows from the observation that the $L^{\underline{th}}$ level substructures, that are comprised of the basic finite element assemblages, represent a small portion of the overall solution time. Consequently, the actual CPU time required to solve a large, isoparametrically square mesh refinement via an HPT will not be significantly affected by the type of finite elements used! Simply put, it does not matter whether 3 or 6 node triangular, 4 or 8 node quadrilateral, etc. elements are used; the HPT solution strategy will yield the same relative speedup and CPU time. It must be reiterated, however, that this is only true for an asymptotically large number of DOF.

Without the use of the HPT, the computational effort for solving h-, p-, and hp-extensions can differ substantially for the same number of DOF. As an example, assuming the same number of DOF per node and $n_1 = \eta$, the relative number of arithmetic operations required to condense out the internal DOF of single h- and p-extended elements is

$$\frac{C_h}{C_p} = \mathcal{O}\left[\frac{1}{(n_1)^2}\right] \tag{5.14}$$

Equation (5.14) clearly shows that these two types of refinement have significantly different computational costs. Turning our attention to h- and hp-extended element/regions, the relative computational effort for performing the condensation process is, in terms of κ and η,

$$\frac{C_h}{C_{hp}} \sim \frac{\text{Equation (3.100)}}{\text{Equation (3.99)}} \tag{5.15}$$

Note that (5.15) was derived with the assumption that the internal DOF of the individual p-extended elements are condensed out before they are assembled. For values of $\eta \leq 15$, it was shown that

$$C_h \geq C_{hp} \tag{5.16}$$

Thus, the relative computational effort involved in condensing/solving the various modes of refinement is, for a fixed number of DOF,

$$C_p > C_h > C_{hp} \tag{5.17}$$

This comparison of the various methods was performed to illustrate that the practice of comparing their relative accuracy on a DOF basis may be misleading. From a pragmatic point of view, it is our opinion that the actual amount of CPU time required to obtain a certain degree of accuracy may be a more relevant form of comparison.

In closing, the HPT has been shown to be computationally efficient and less demanding of memory resources. From a more philosophical perspective, the HPT solution strategy also provides:

1. Localized "error-trapping". This occurs because a given DOF, as a result of the hierarchical substructuring process, will have a compacted column height. Consequently, the reduced coupling with other DOF diminishes the direct influence that a finite element approximation error can have on the rest of the model.

2. A means with which to "telescope" into a physical anomaly or analytical singularity by grafting another localized HPT onto the previous one. This is a logical, efficient way of traversing from "coarse" to "fine" scales of model definition.

REFERENCES:

1. S. Adjerid and J. Flaherty, "A Moving-Mesh Finite Element Method with Local Refinement for Parabolic Partial Differential Equations", Computer Methods In Applied Mechanics and Engineering, 55, 1986, pp 3-26.

2. I. Babuska, J. Chandra, and J. Flaherty, "Adaptive Computational Methods for Partial Differential Equations", SIAM, Philadelphia, PA, 1983.

3. I. Babuska, O.C. Zienkiewicz, E. Arantes e Oliveira, J.R. Gago, and K. Morgan, "Accuracy Estimates and Adaptivity for Finite Elements", Wiley, London, 1986.

4. "Special Issue: ADAPTIVE METHODS", S.N. Atluri, T. Belytschko, J.T. Oden, J.N. Reddy, editors, Computer Methods In Applied Mechanics and Engineering, Vol. 55, 1986.

5. J. Padovan, D. Gute, and K. Johnson, "Hierarchical Poly Tree Computer Architectures Defined by Computational Multidisciplinary Mechanics", Computers and Structures, Vol. 32, No. 5, pp 1133-1163, 1989.

6. J. Padovan and D. Gute, "Multi-Level Hierarchical Poly Tree Computer Architectures", Computers and Structures, (in press)

7. K. Bathe, "Finite Element Procedures in Engineering Analysis", Prentice Hall, Inc., Englewood Cliffs, New Jersey 07632, 1982.

8. ADINA®, Report Ard. 84-1, Users Manual. ADINA R&D, Inc., 71 Elton Avenue, Watertown, MA 02171, USA (1984).

9. O.C. Zienkiewicz and R.L. Taylor, "The Finite Element Method, 4th ed., Volume 1 Basic Formulation and Linear Problems", McGraw-Hill Book Co., 1989.

10. I. Katz, A. Peano, and M. Rossow, "Nodal Variables for Complete Finite Elements of Arbitrary Polynomial Order", Computer Math. Appl., 4 (1978), pp 85-112.

11. I. Babuska, B.A. Szabo, and I.N. Katz, "The p-Version of the Finite Element Method", SIAM J. Numer. Anal., 18 (1981) 515-545.

12. B.A. Szabo, "Control of the Errors of Discretization and Idealization in Finite Element Analysis", NASA Workshop on Computational Structural Mechanics, NASA Langley Research Center, Nov. 18-20, 1987.

13. I. Babuska and M.R. Dorr, "Error estimates for the Combined h and p Versions of the Finite Element Method", Tech. Rept. BN-951, Institute for Physical Sciences and Technology, University of Maryland, College Park, MD, 1980

14. C. Farhat and E. Wilson, "A Parallel Active Column Equation Solver", Computers and Structures, Vol. 28, No. 2, pp 289-304, 1988.

FIGURE CAPTIONS

2.1) Schematic of Multilevel Hierarchical Poly Tree (HPT)

2.2) Schematic of Root-Branch System

2.3) Finite Element Model of Transverse Fluid Flow Within Tubular Bundles

2.4) Finite Element Model of An Arch

3.1) Schematic of a Multilevel HPT for a Localized Mesh Refinement

3.2) h-Extended 4-Node Quadrilateral Element

3.3) Schematic of Algorithmic Steps for an h-Extended Element/Region

3.4) Dissemination of an h-Extended Element/Region into $(K_2)^2$ Substructures

3.5) Assemblage of Statically Condensed Substructures

3.6) Schematic of the Algorithmic Steps Associated with the First Level of the HPT

3.7) Theoretical Speedups for Sequential Two-Level HPTs Applied to an h-Extension (K_2 = 2,3, and 4)

3.8) Speedup for Sequential, Two-Level HPT Applied to an h-Extension (ρ = 1, K_2 = 2, L = 2)

3.9) Speedup for Sequential, Two-Level HPT Applied to an h-Extension (ρ = 2, K_2 = 2, L = 2)

3.10) Speedup for Sequential, Two-Level HPT Applied to an h-Extension (ρ = 1, K_2 = 3, L = 2)

3.11) Speedup for Sequential, Two-Level HPT Applied to an h-Extension (ρ = 2, K_2 = 3, L = 2)

3.12) Fractional Number of Stiffness Matrix Elements That Must Be Stored When Using an HPT for an h-Extension

3.13) Diagram of "Telescoping" Technique

3.14) p-Extended 4-Node Quadrilateral Element

3.15) Static Condensation of a p-Extended 4-Node Quadrilateral Element

TABLE CAPTIONS

3.1) Speedups for Two-Level, Sequential HPT Applied to an h-Extension with $K_2 = 2$

3.2) Speedups for Two-Level, Sequential HPT Applied to an h-Extension with $K_2 = 3$

3.3) Speedups for Three-Level, Sequential HPT Applied to an h-Extension with $K_2 = K_3 = 2$

4.1) Effective Speedups and Superlinearities for Two-Level, Parallel HPT Applied to an h-Extension with $K_2 = 2,3$ and $\rho = 1$

4.2) Effective Speedups and Superlinearities for Two-Level, Parallel HPT Applied to an h-Extension With $K_2 = 2,3$ and $\rho = 2$

4.3) Effective Speedups and Superlinearities for Three-Level, Parallel HPT Applied to an h-Extension with $K_2 = K_3 = 2$

4.4) Effective Speedups and Superlinearities for Three-Level, Parallel HPT with One-Level of Sequentialism ($\mathcal{L} = 1$) Applied to an h-Extension ($K_2 = K_3 = 2$)

APPENDIX A

The appendix is comprised of four figures which illustrate the numbering schemes employed for the substructuring primitives used in the development of the HPT. Note that

1) The "internal" degrees of freedom (DOF) are numbered first; and,

2) The "external" DOF lie on the periphery of each substructuring primitive.

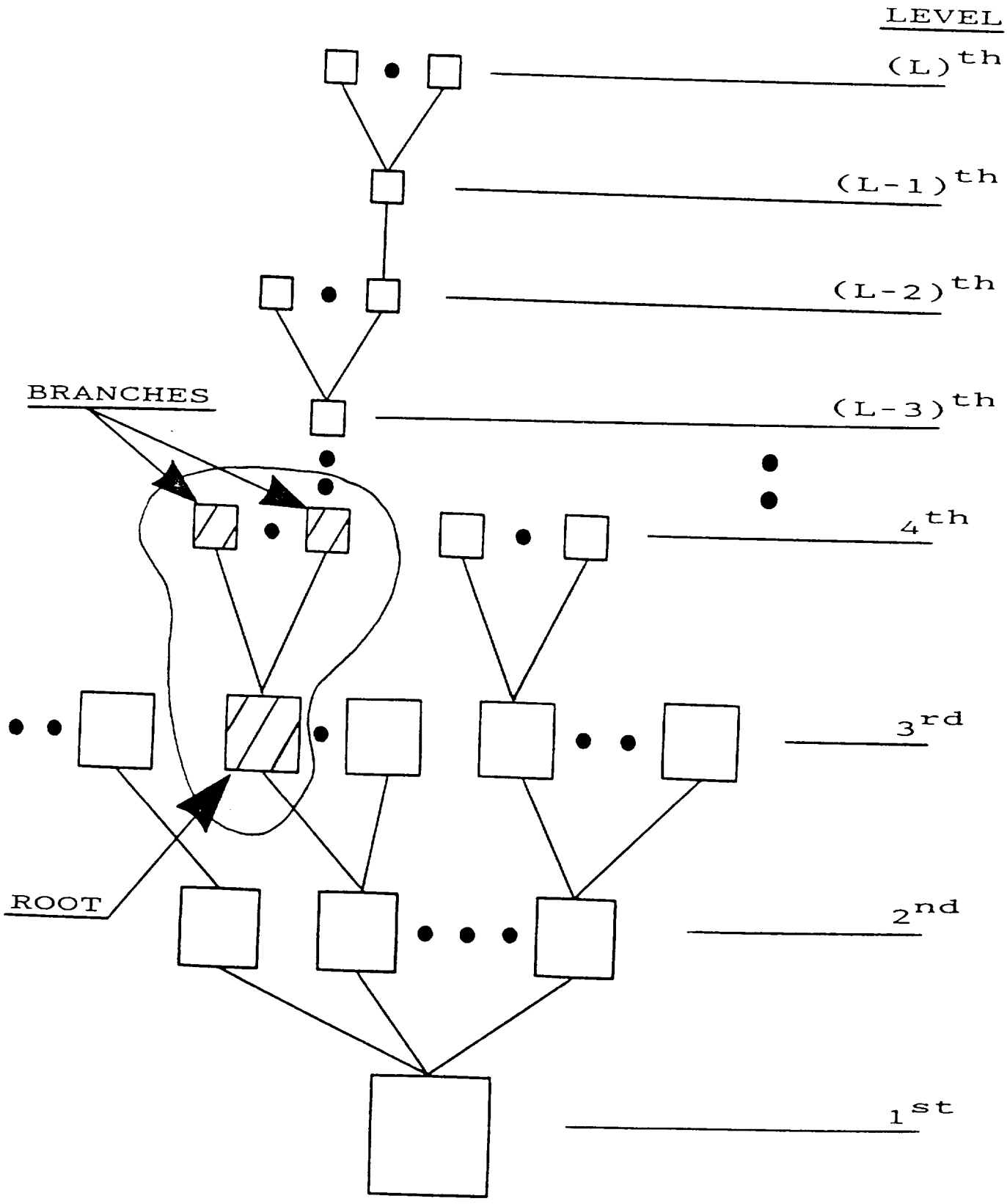


FIGURE 2.1

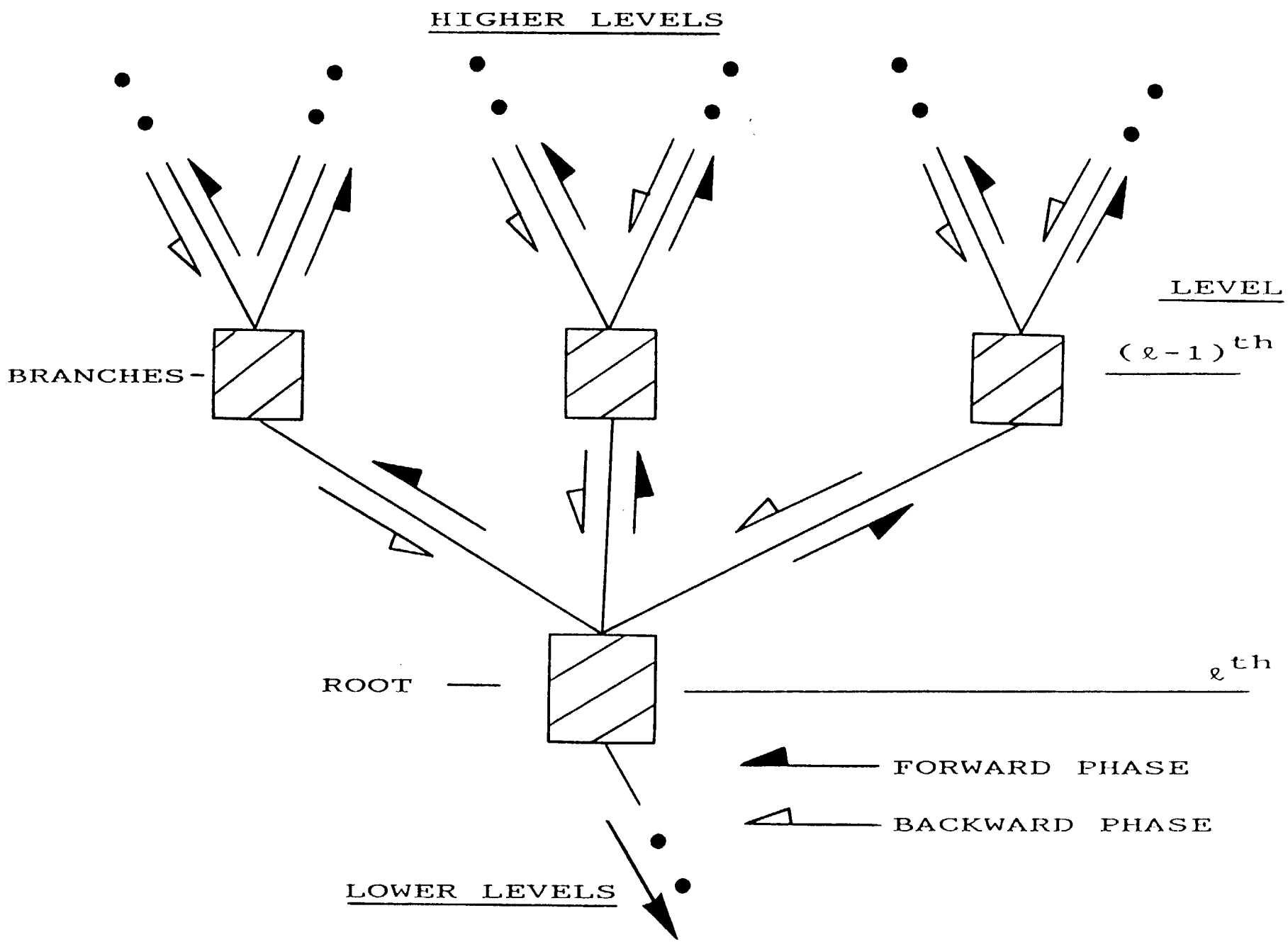

HIGHER LEVELS
LEVEL
$(\ell-1)^{th}$
BRANCHES
ROOT
ℓ^{th}
FORWARD PHASE
BACKWARD PHASE
LOWER LEVELS

FIGURE 2.2

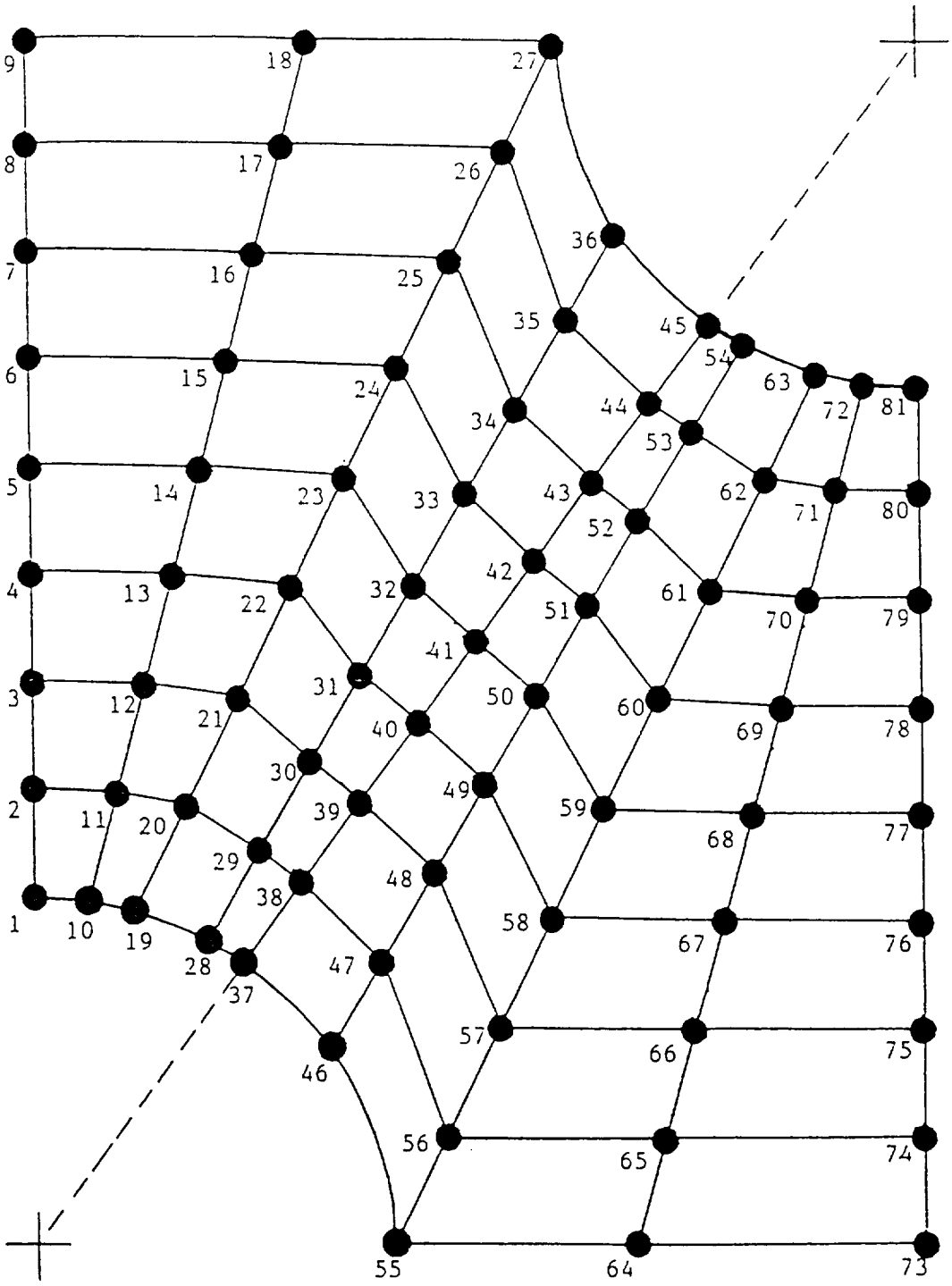

9
18
27
8
17
26
36
7
16
25
35
45
54
6
15
24
34
44
53
63
72
81
5
14
23
33
43
52
62
71
80
4
13
22
32
42
51
61
70
79
3
12
21
31
41
50
60
69
78
2
11
20
30
40
49
59
68
77
1
10
19
28
39
48
58
67
76
29
38
47
57
66
75
37
46
56
65
74
55
64
73

FIGURE 2.3

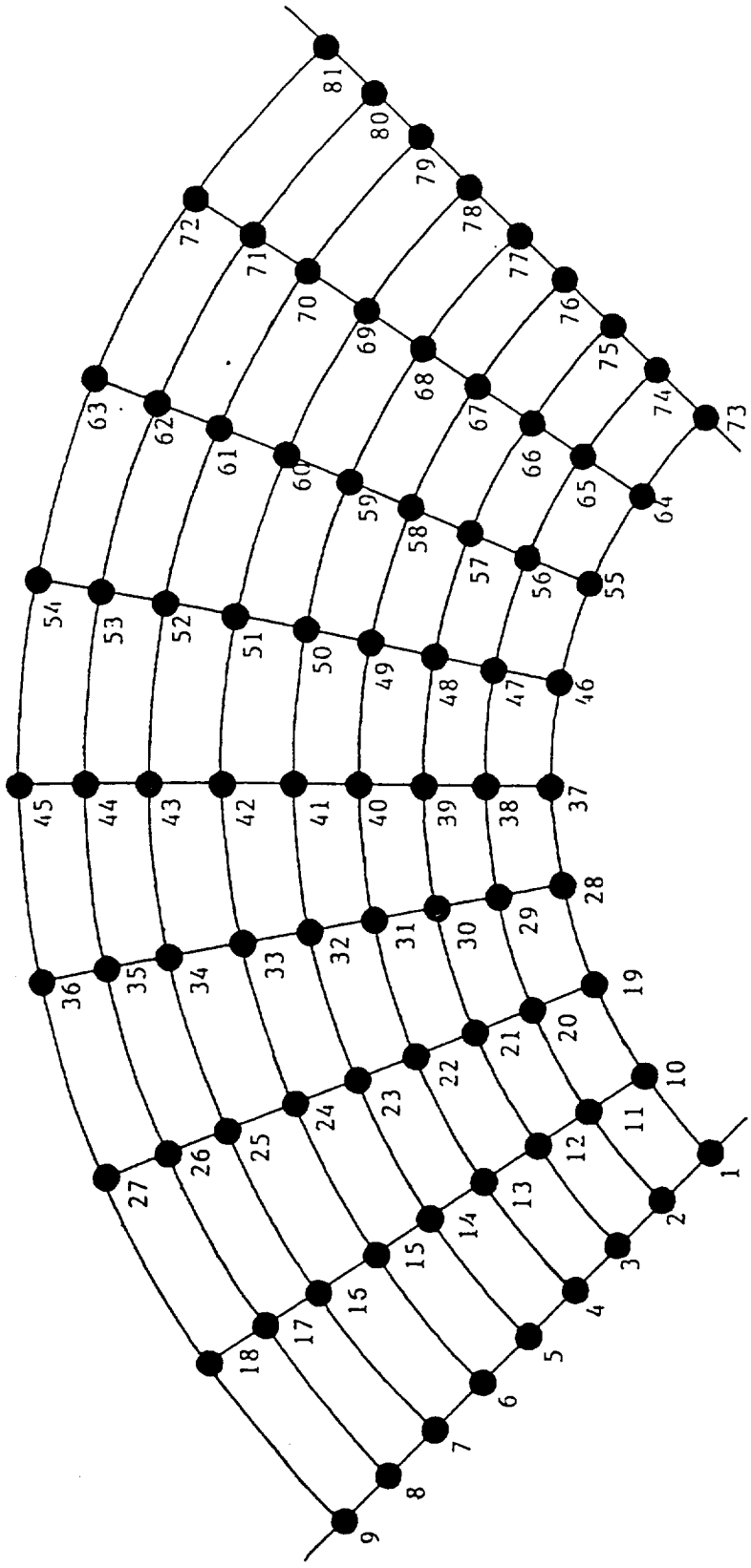

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81

FIGURE 2.4

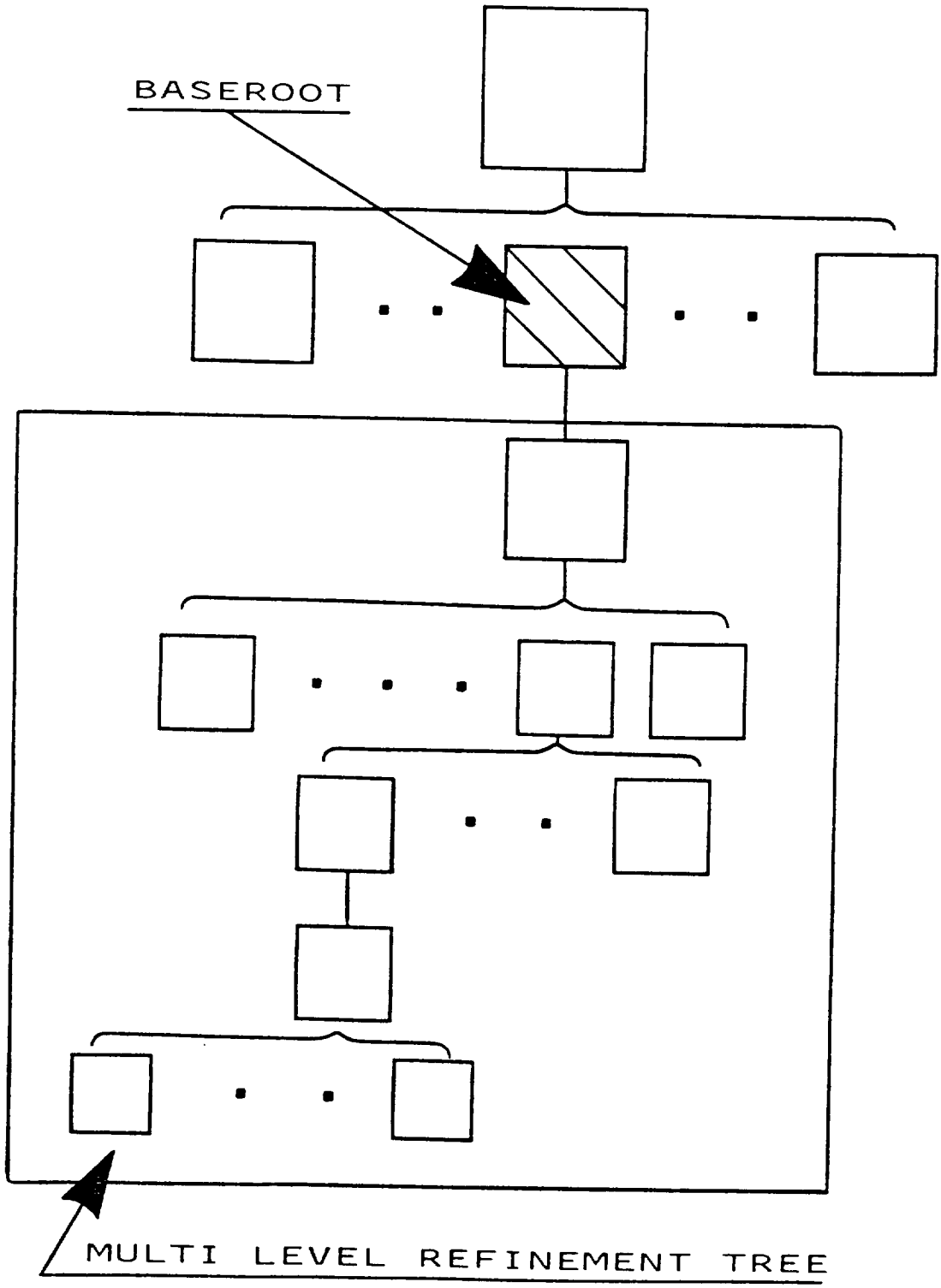


FIGURE 3.1

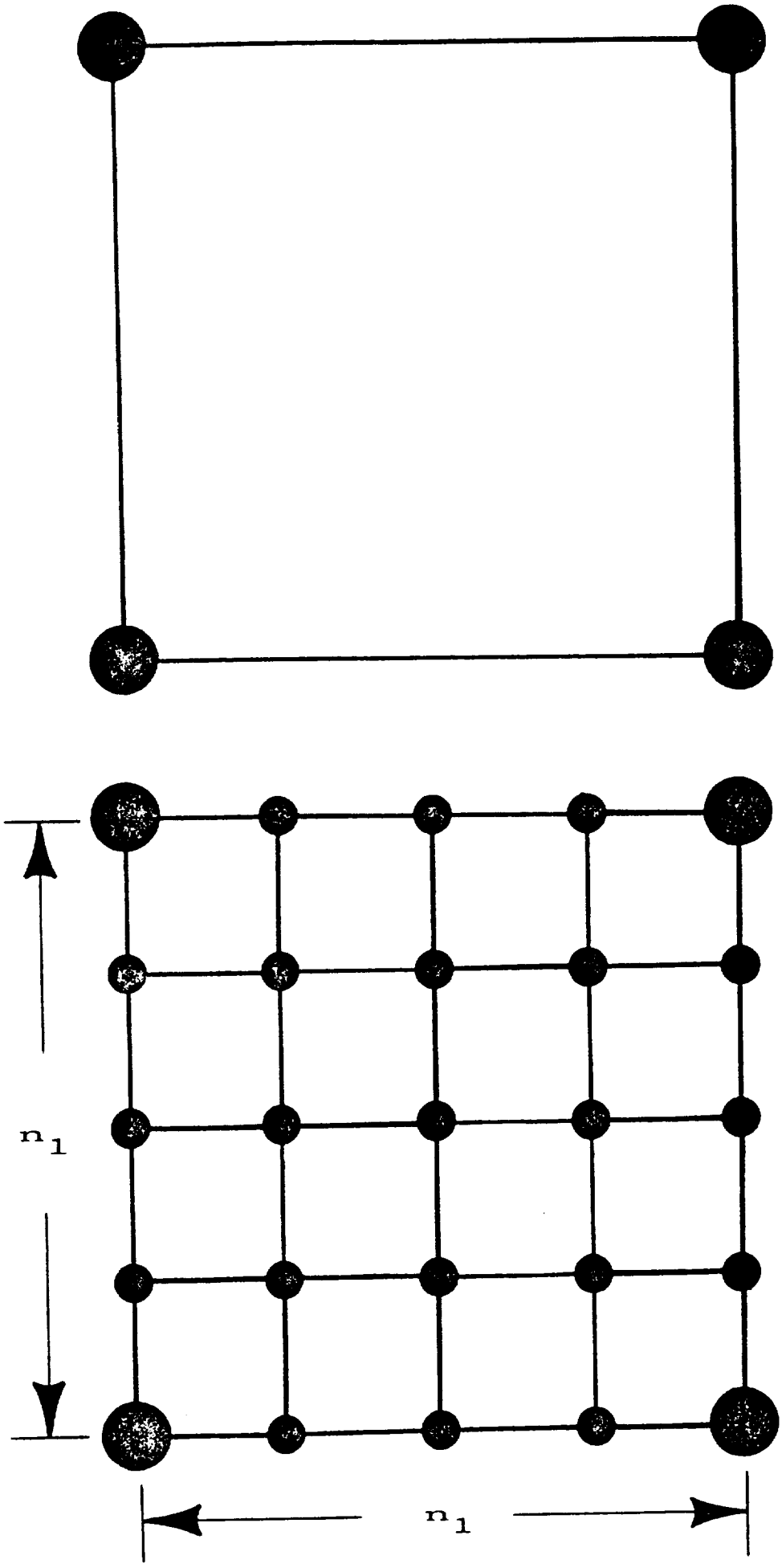


FIGURE 3.2

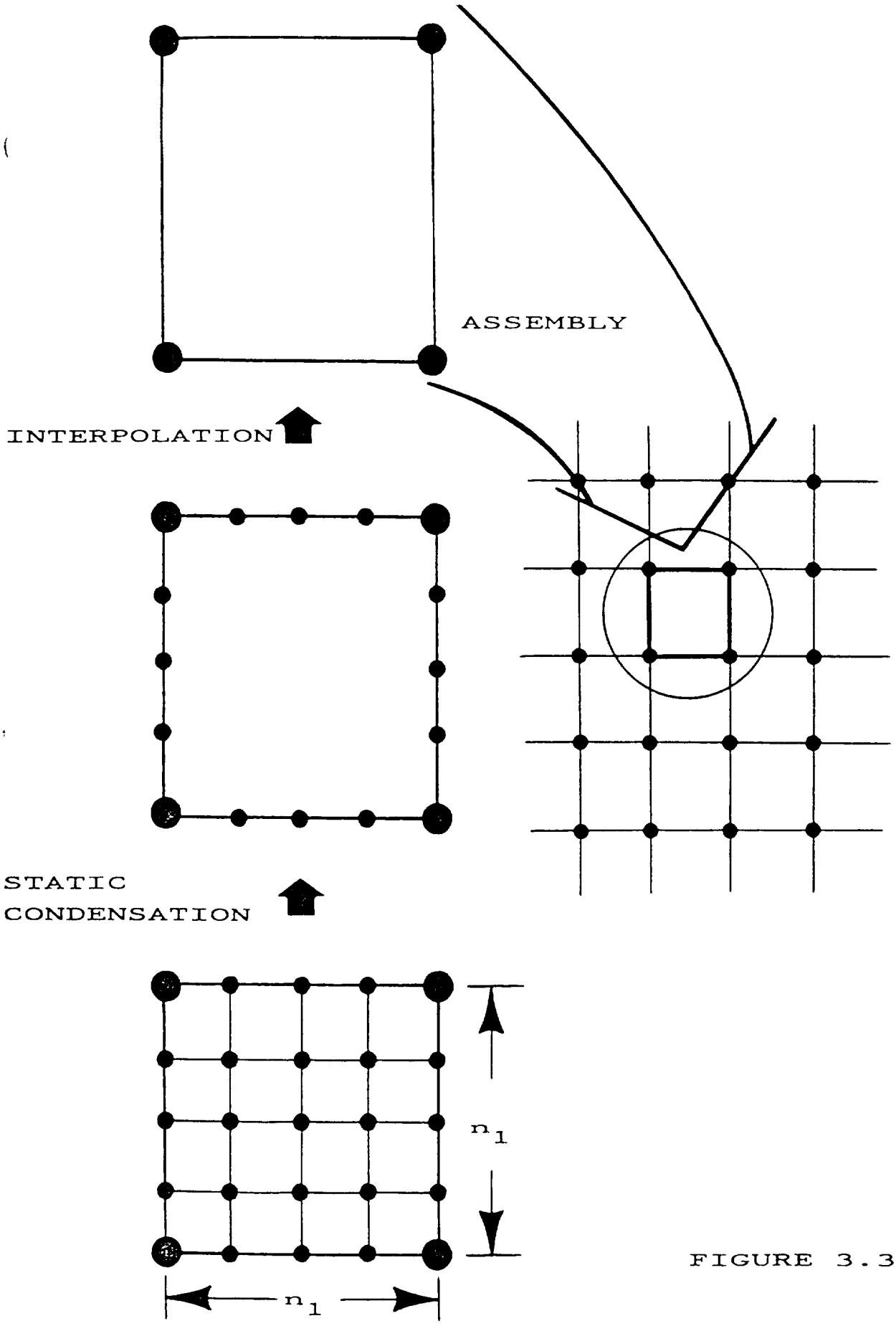


FIGURE 3.3

$$n_2 = \frac{n_1 + K_2 - 1}{K_2}$$

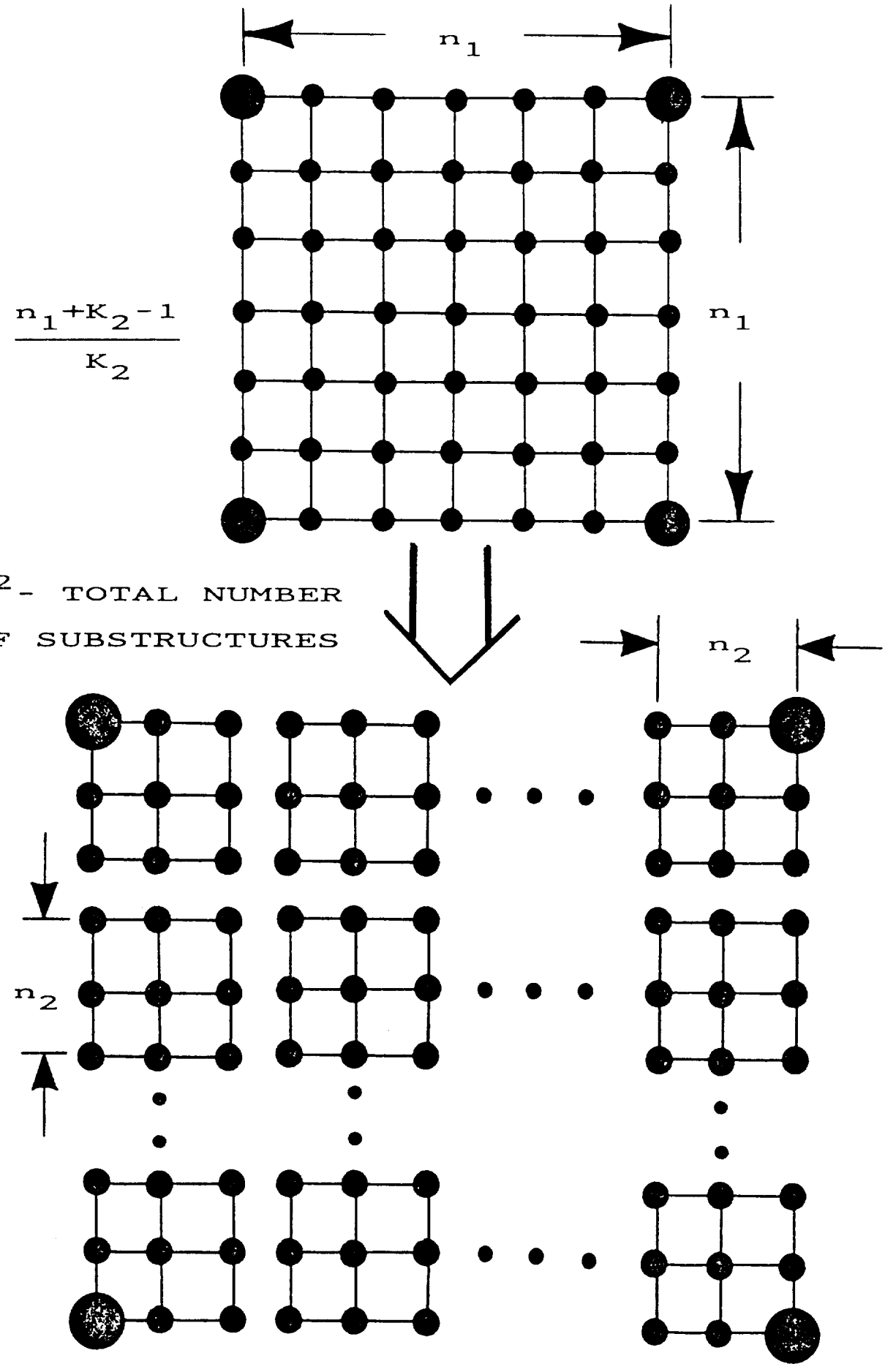


FIGURE 3.4

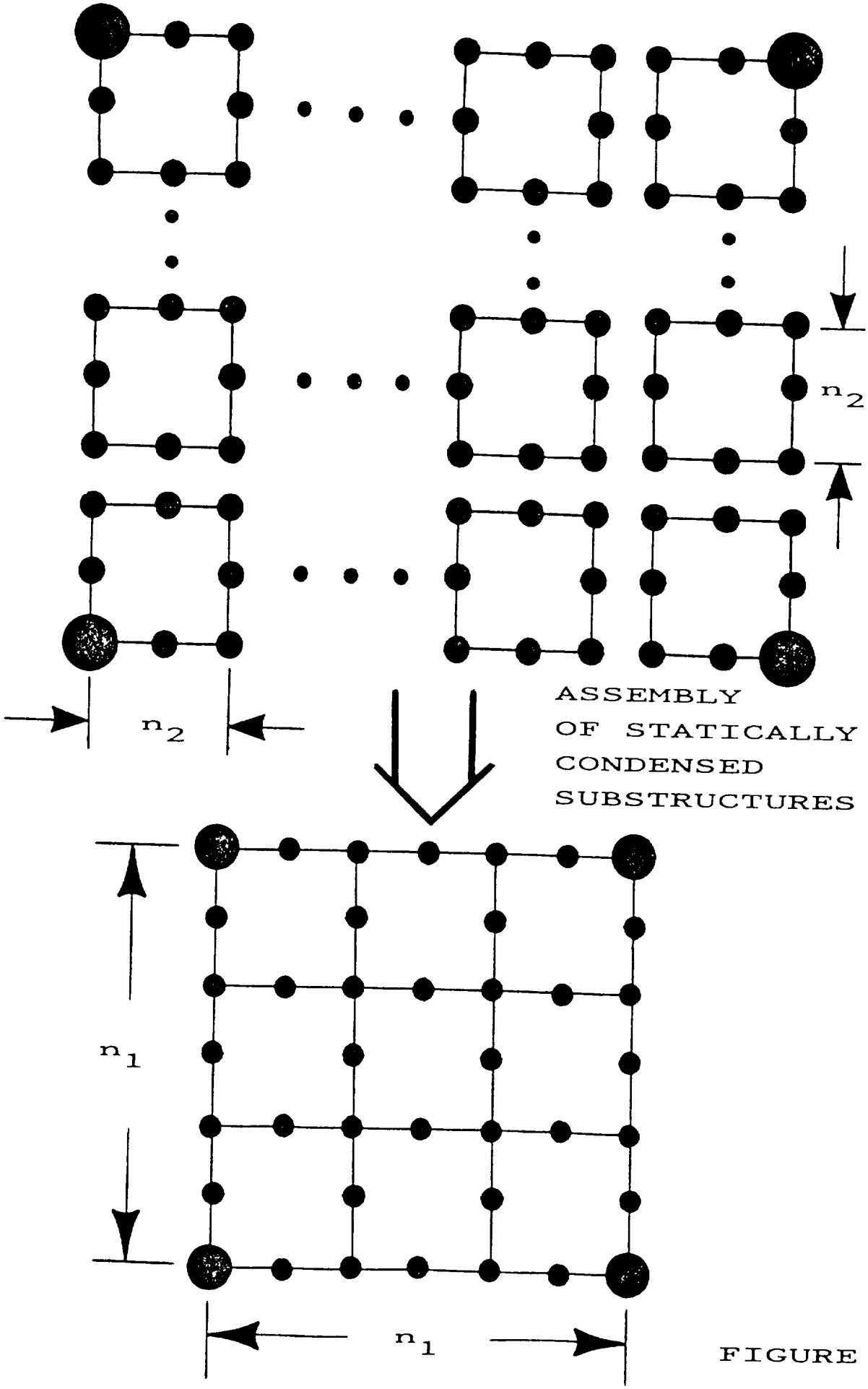

n_2
n_2
ASSEMBLY
OF STATICALLY
CONDENSED
SUBSTRUCTURES
n_1
n_1

FIGURE 3.5

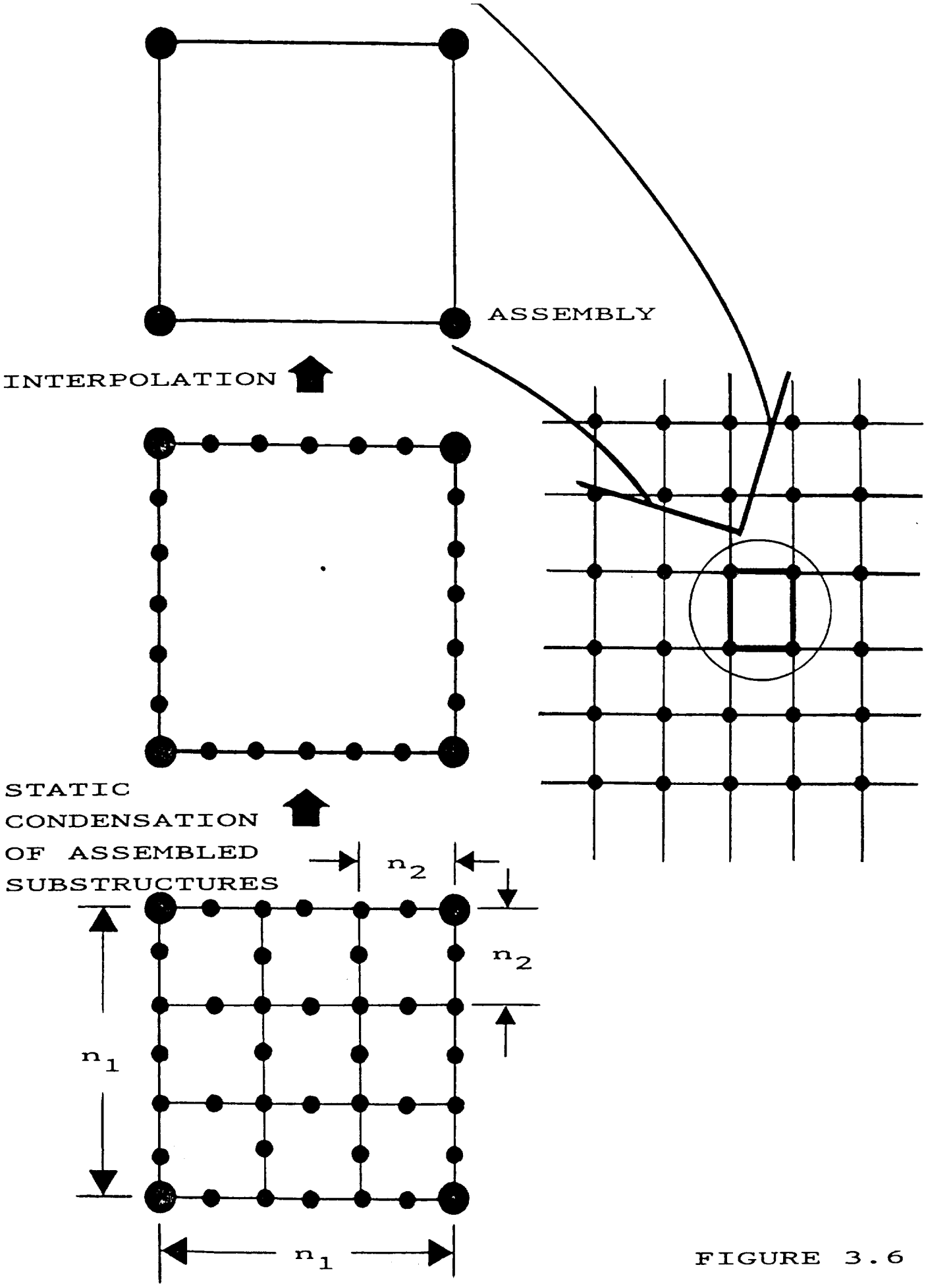

ASSEMBLY
INTERPOLATION
STATIC
CONDENSATION
OF ASSEMBLED
SUBSTRUCTURES
n_2
n_2
n_1
n_1

FIGURE 3.6

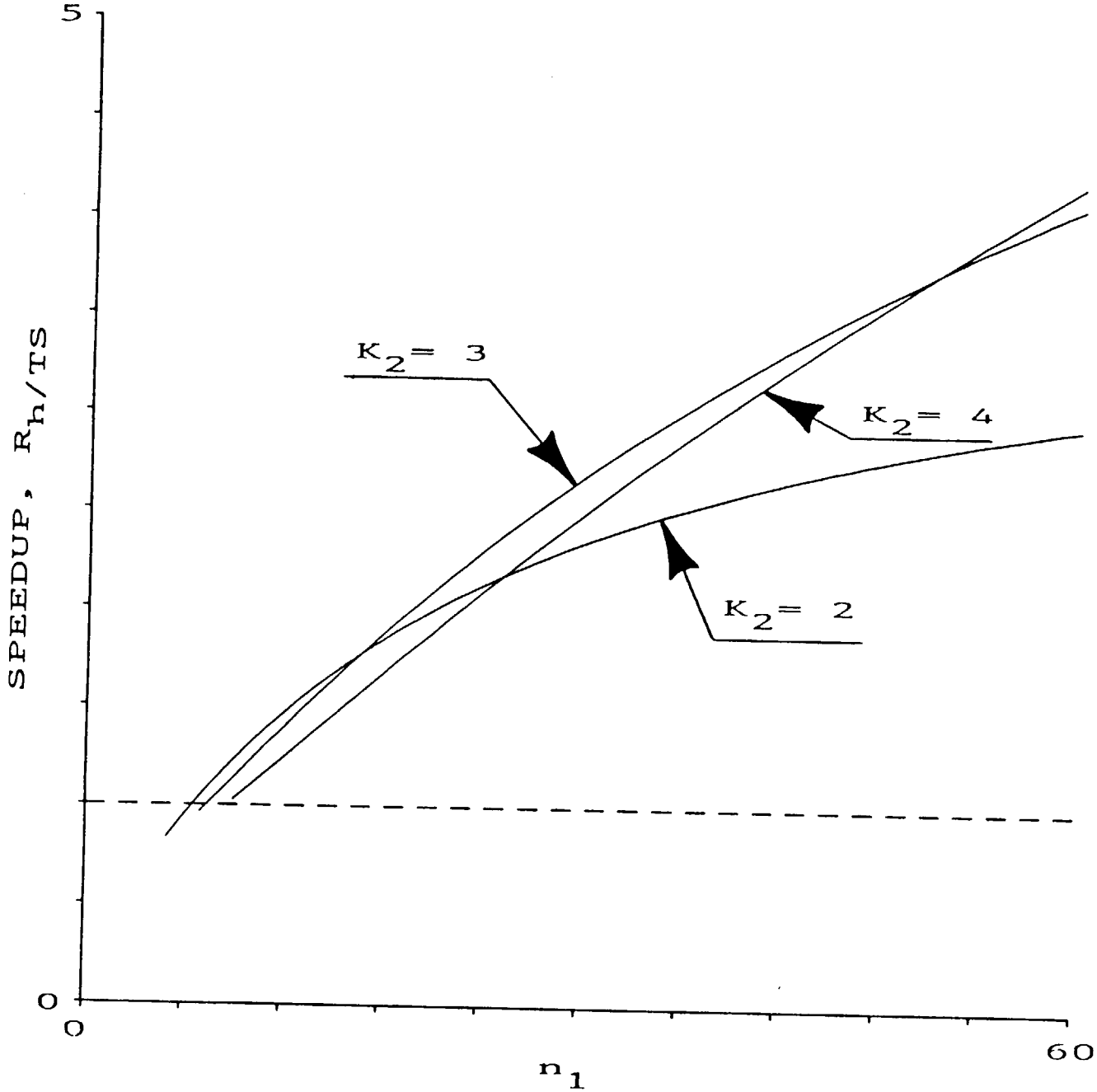


FIGURE 3.7

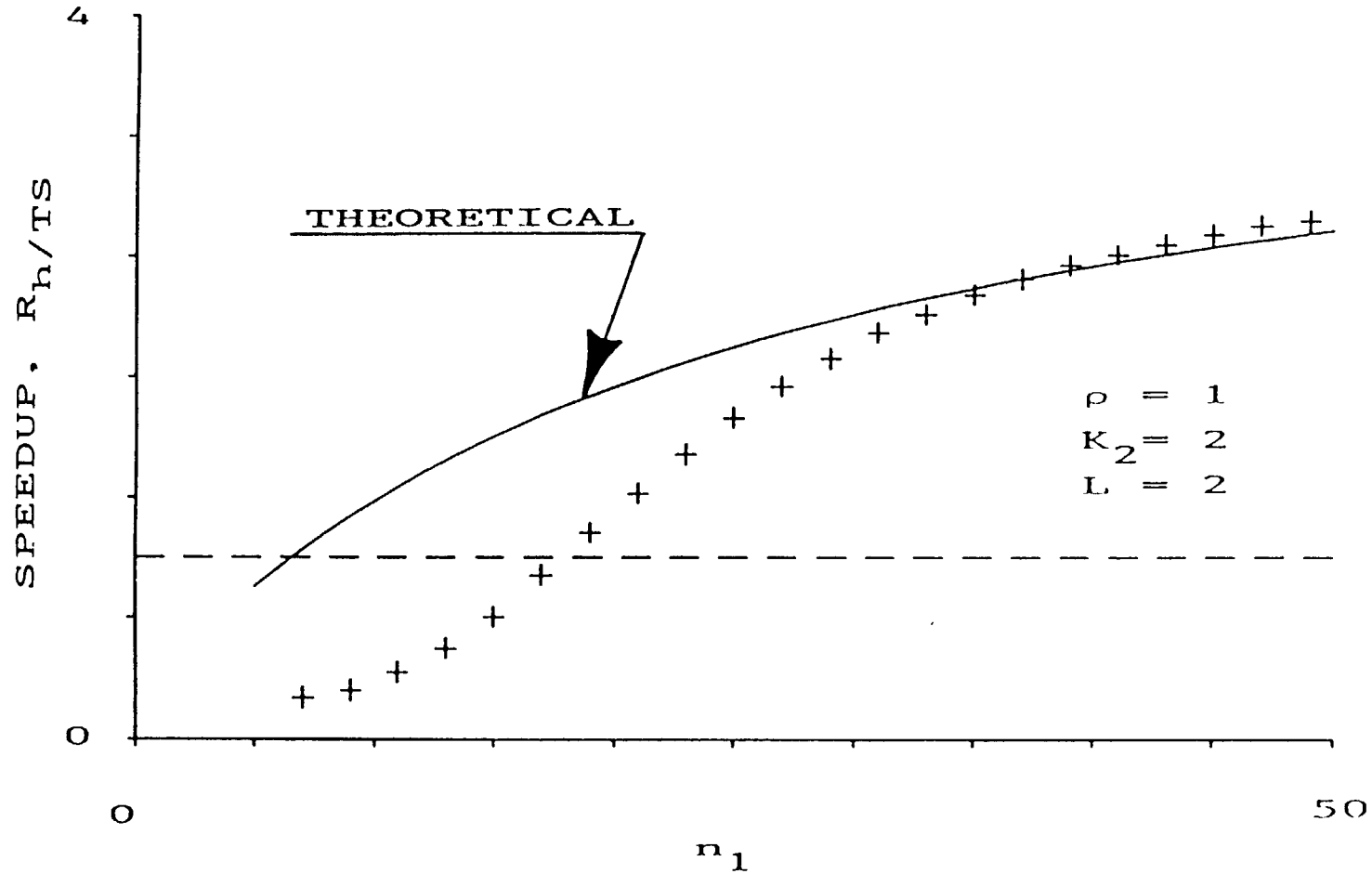


FIGURE 3.8

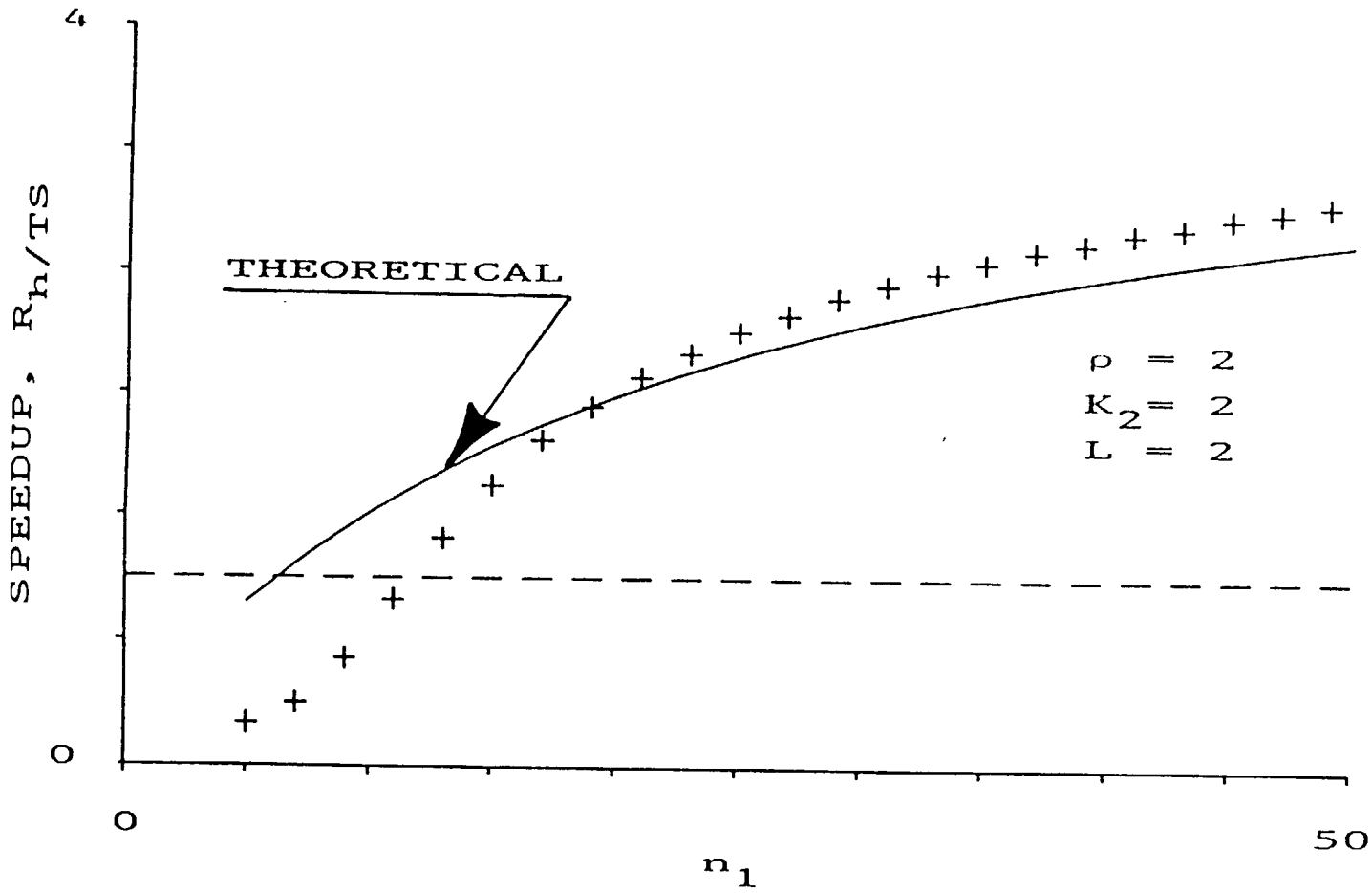


FIGURE 3.9

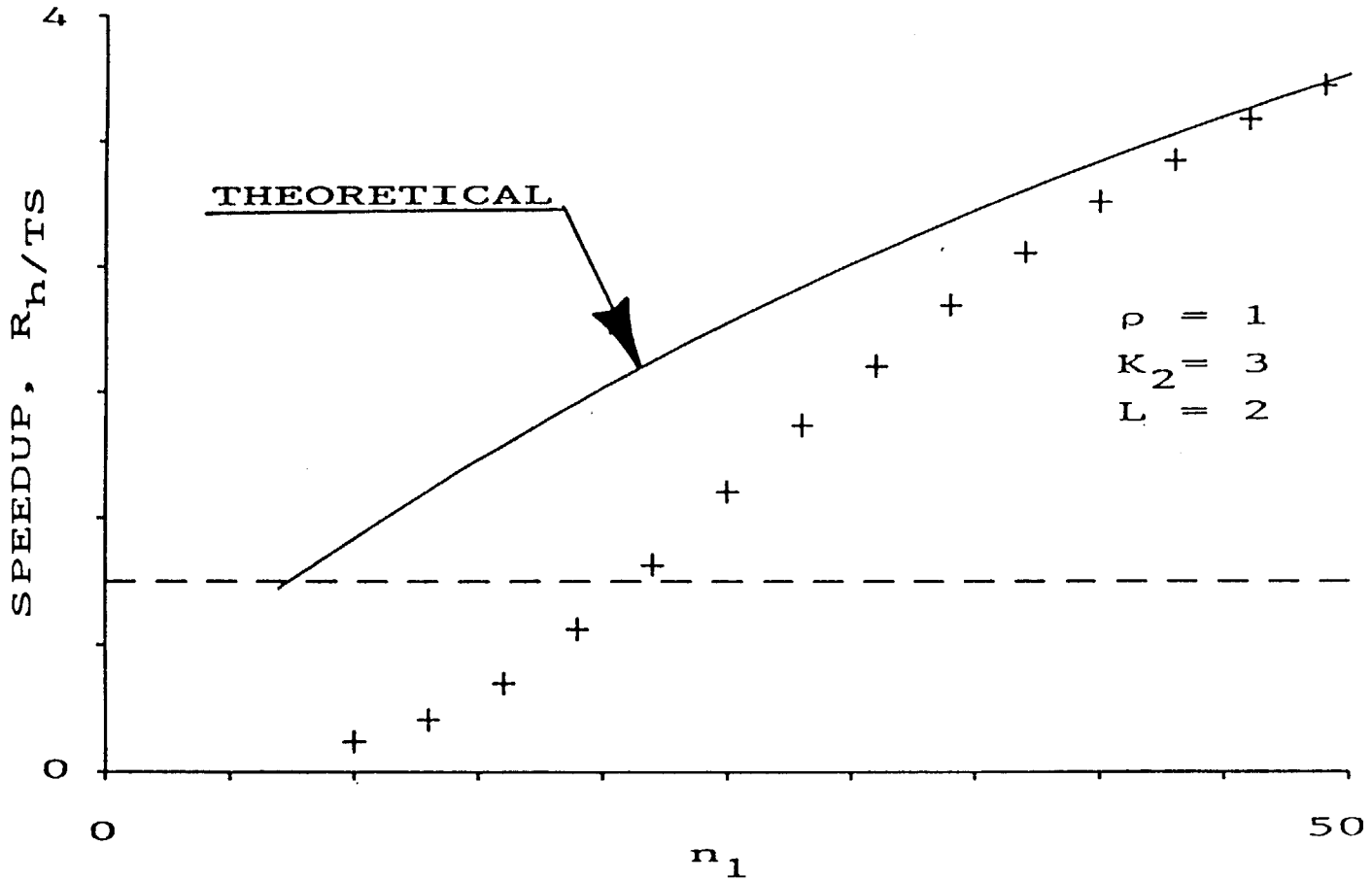


FIGURE 3.10

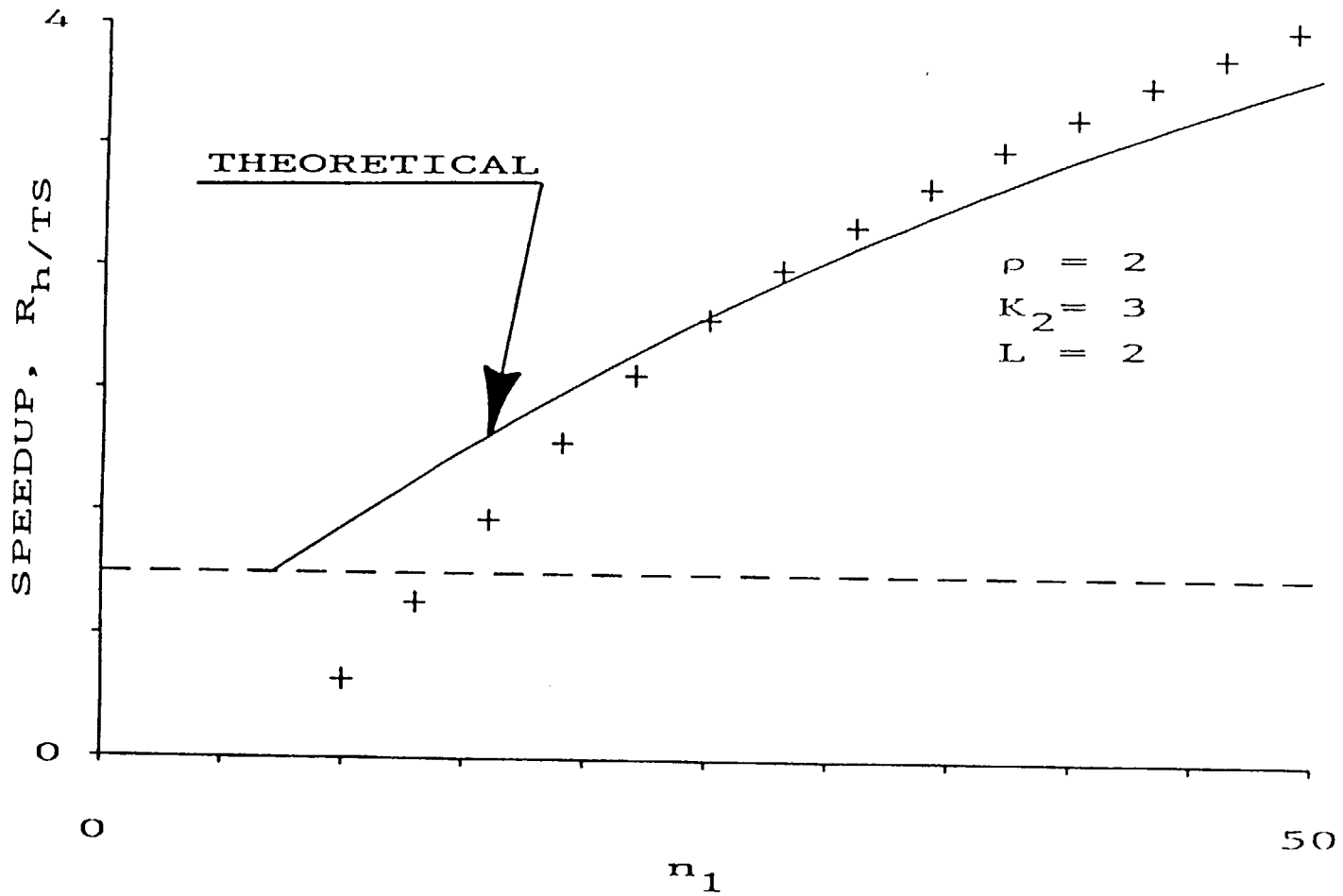


FIGURE 3.11

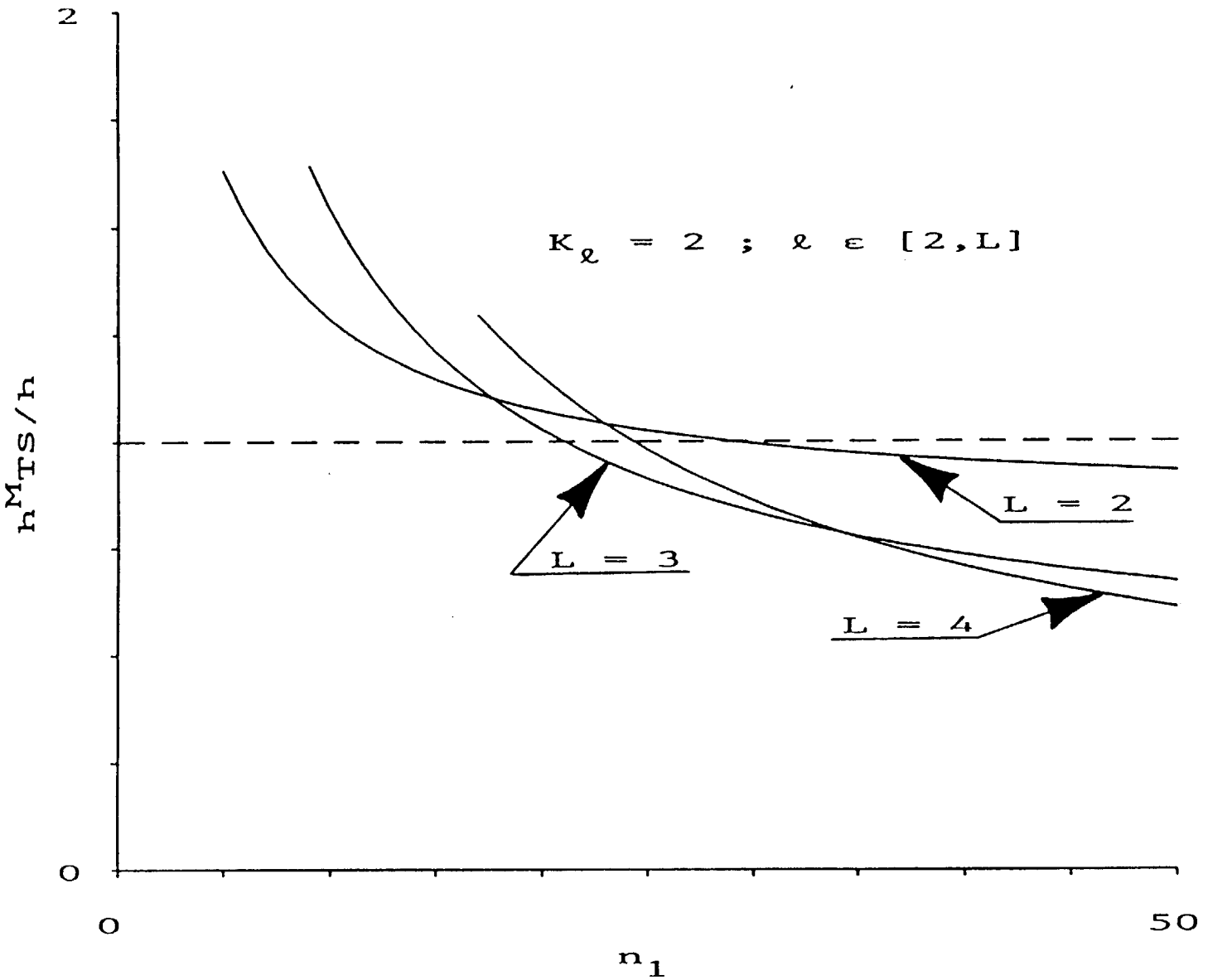


FIGURE 3.12

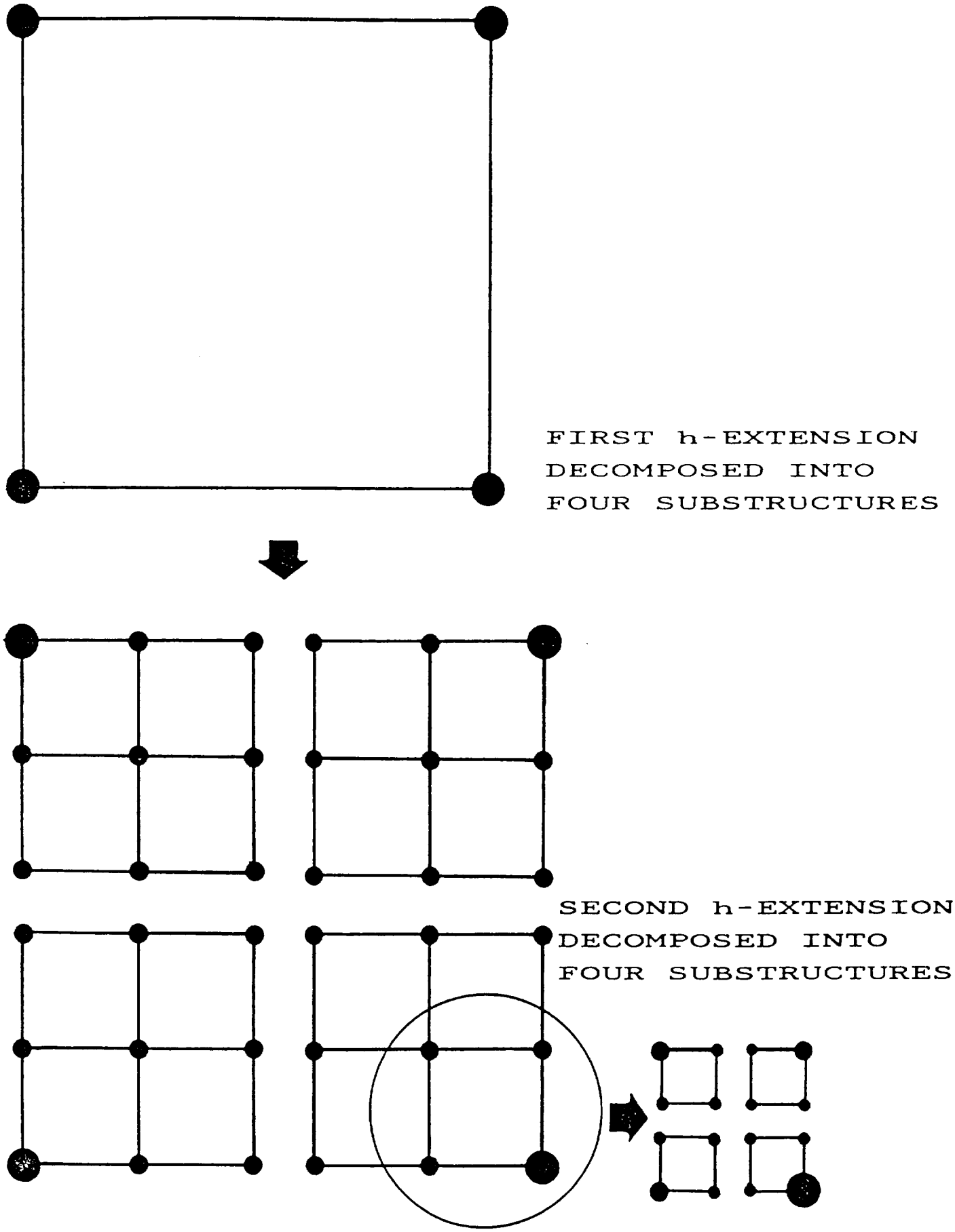

FIRST h-EXTENSION
DECOMPOSED INTO
FOUR SUBSTRUCTURES
SECOND h-EXTENSION
DECOMPOSED INTO
FOUR SUBSTRUCTURES

FIGURE 3.13

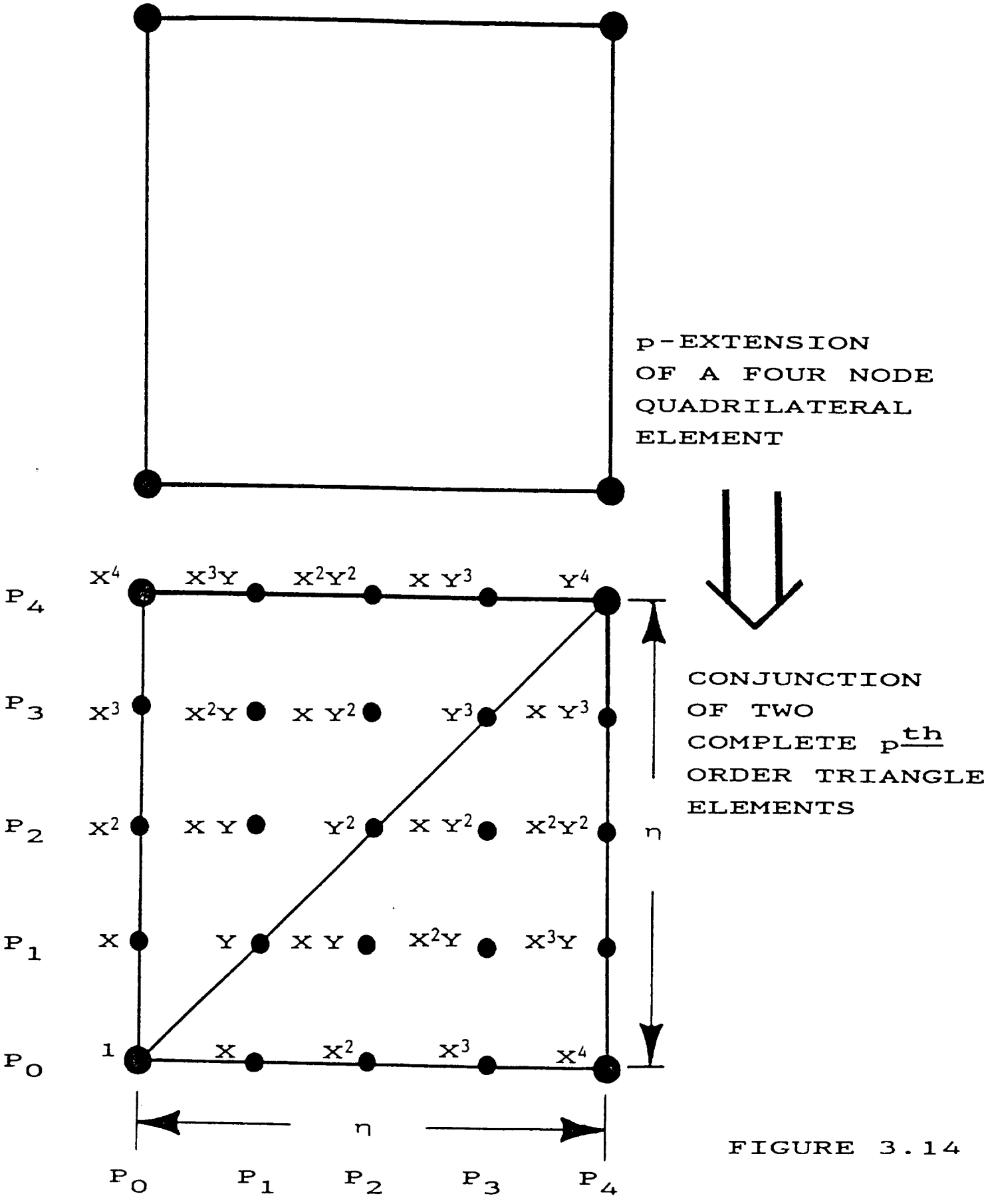
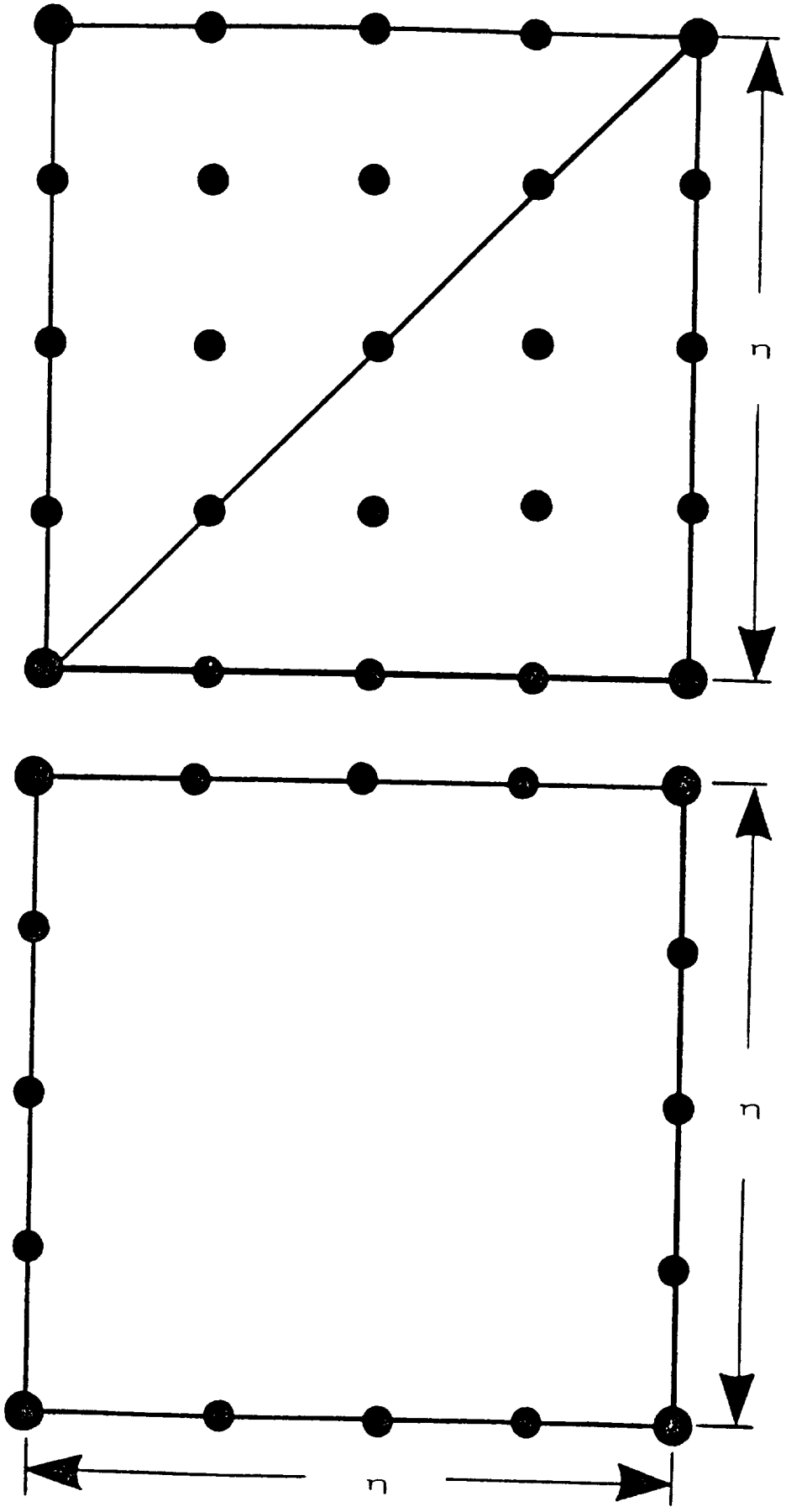
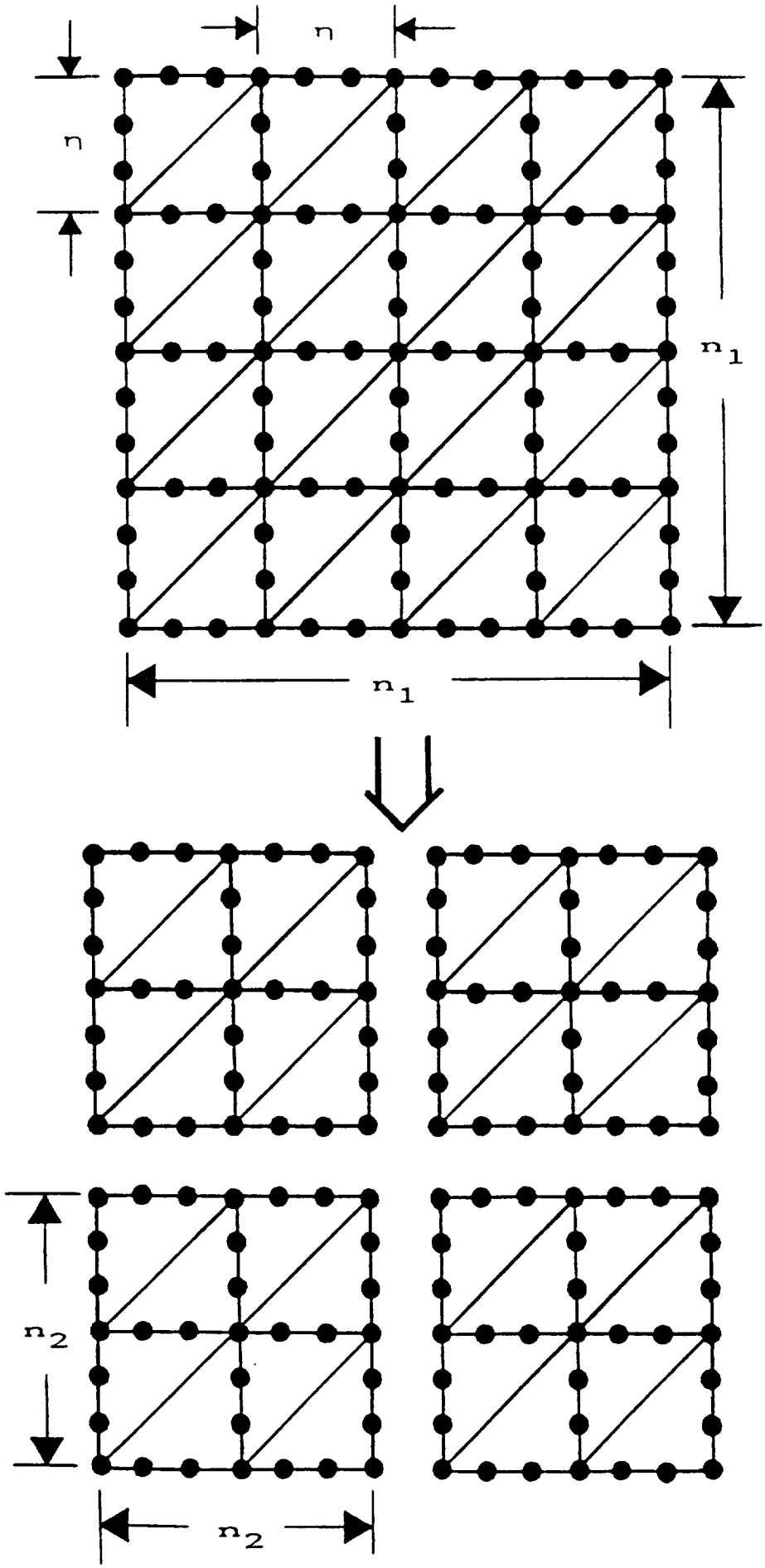

p-EXTENSION
OF A FOUR NODE
QUADRILATERAL
ELEMENT
CONJUNCTION
OF TWO
COMPLETE p^th
ORDER TRIANGLE
ELEMENTS
P4
P3
P2
P1
P0
X4 X3Y X2Y2 X Y3 Y4
X3 X2Y X Y2 Y3 X Y3
X2 X Y Y2 X Y2 X2Y2
X Y X Y X2Y X3Y
1 X X2 X3 X4
η
η
P0 P1 P2 P3 P4

FIGURE 3.14



STATICALLY
CONDENSED
p-EXTENDED
ELEMENT

FIGURE 3.15



ASSEMBLAGE OF CONDENSED p-EXTENDED ELEMENTS

SUBSTRUCTURAL DECOMPOSITION

FIGURE 3.16

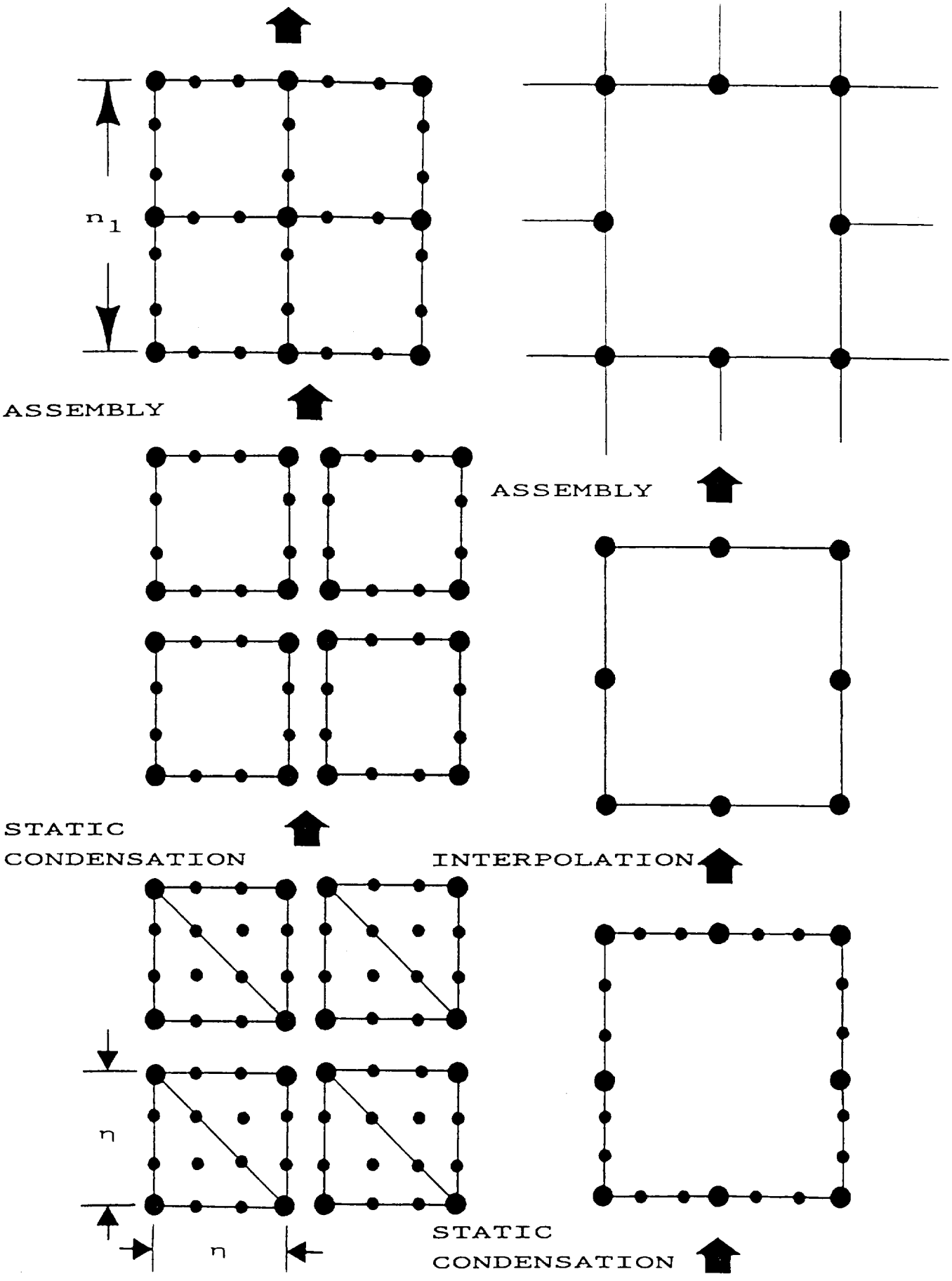

n_1
ASSEMBLY
ASSEMBLY
STATIC
CONDENSATION
INTERPOLATION
n
n
STATIC
CONDENSATION

FIGURE 3.17

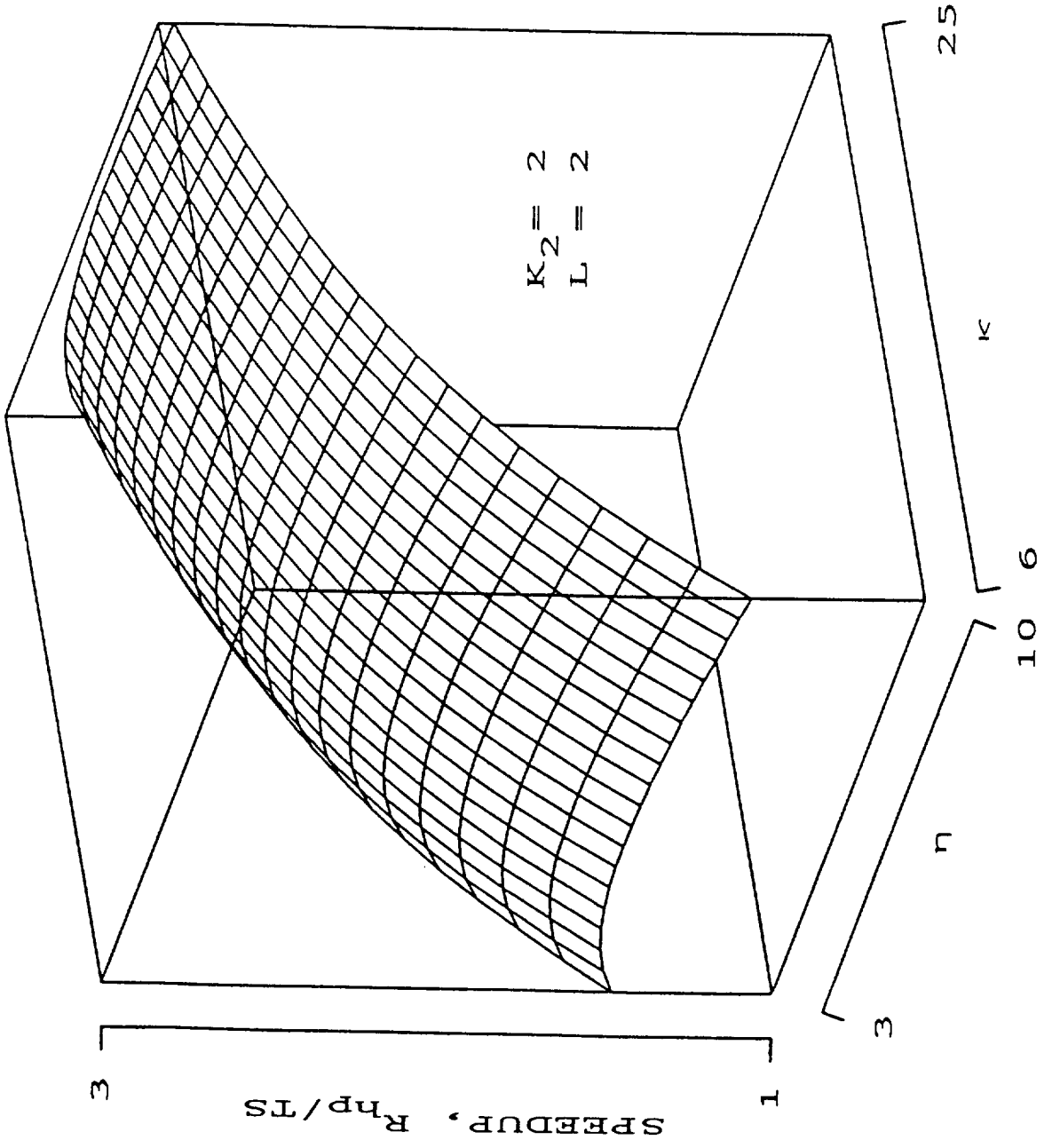


FIGURE 3.18a

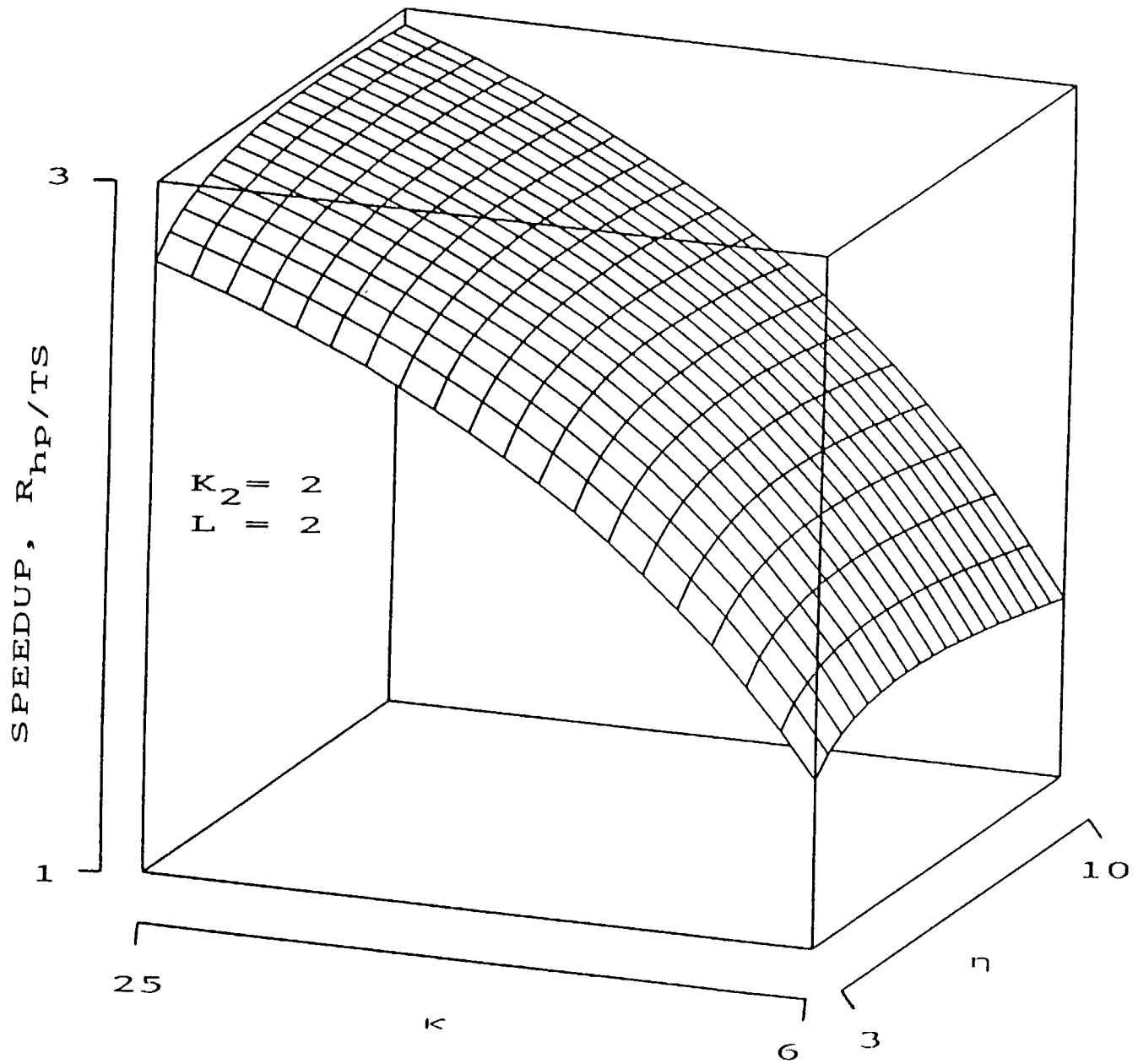


FIGURE 3.18b

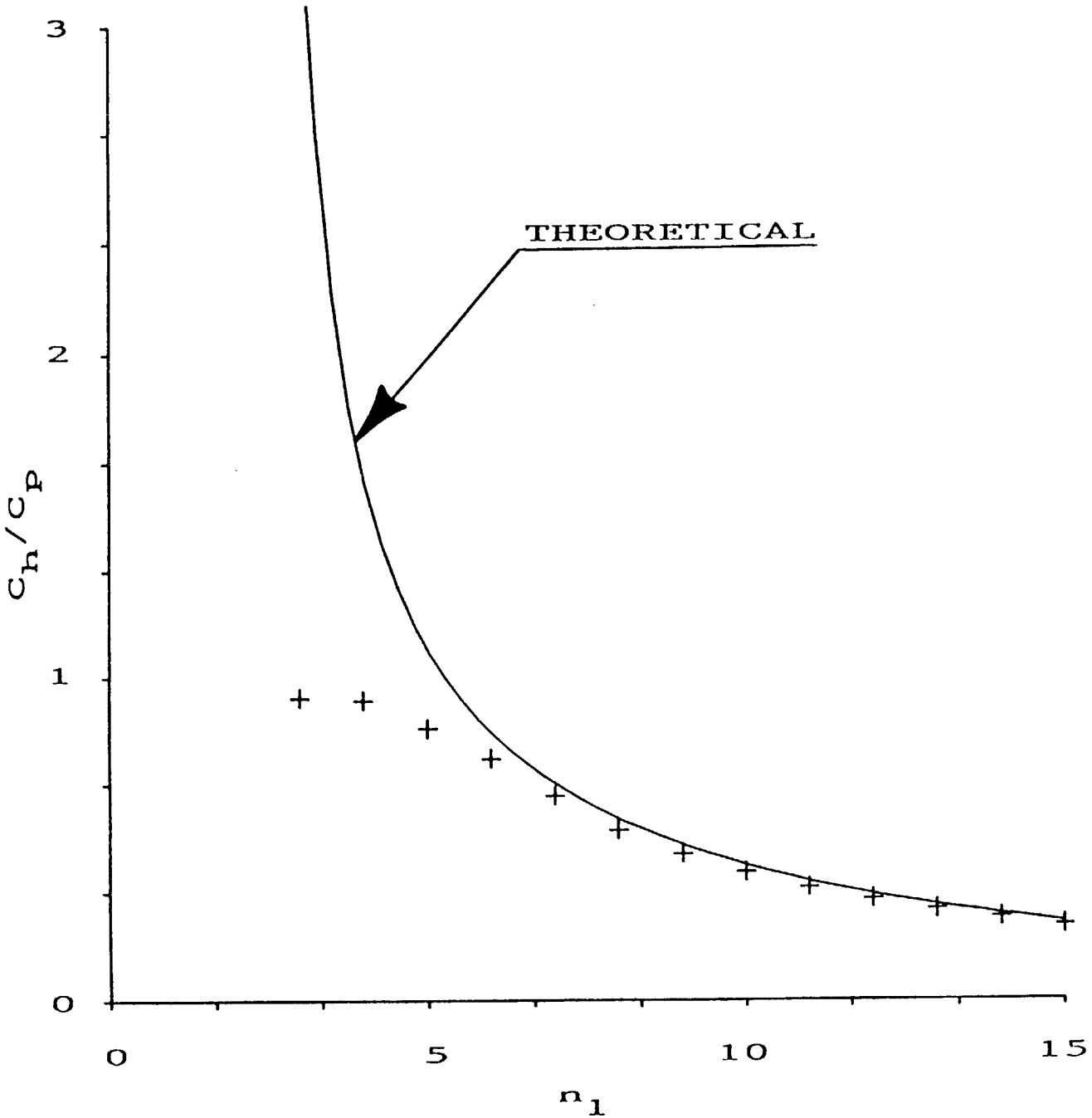


FIGURE 3.19

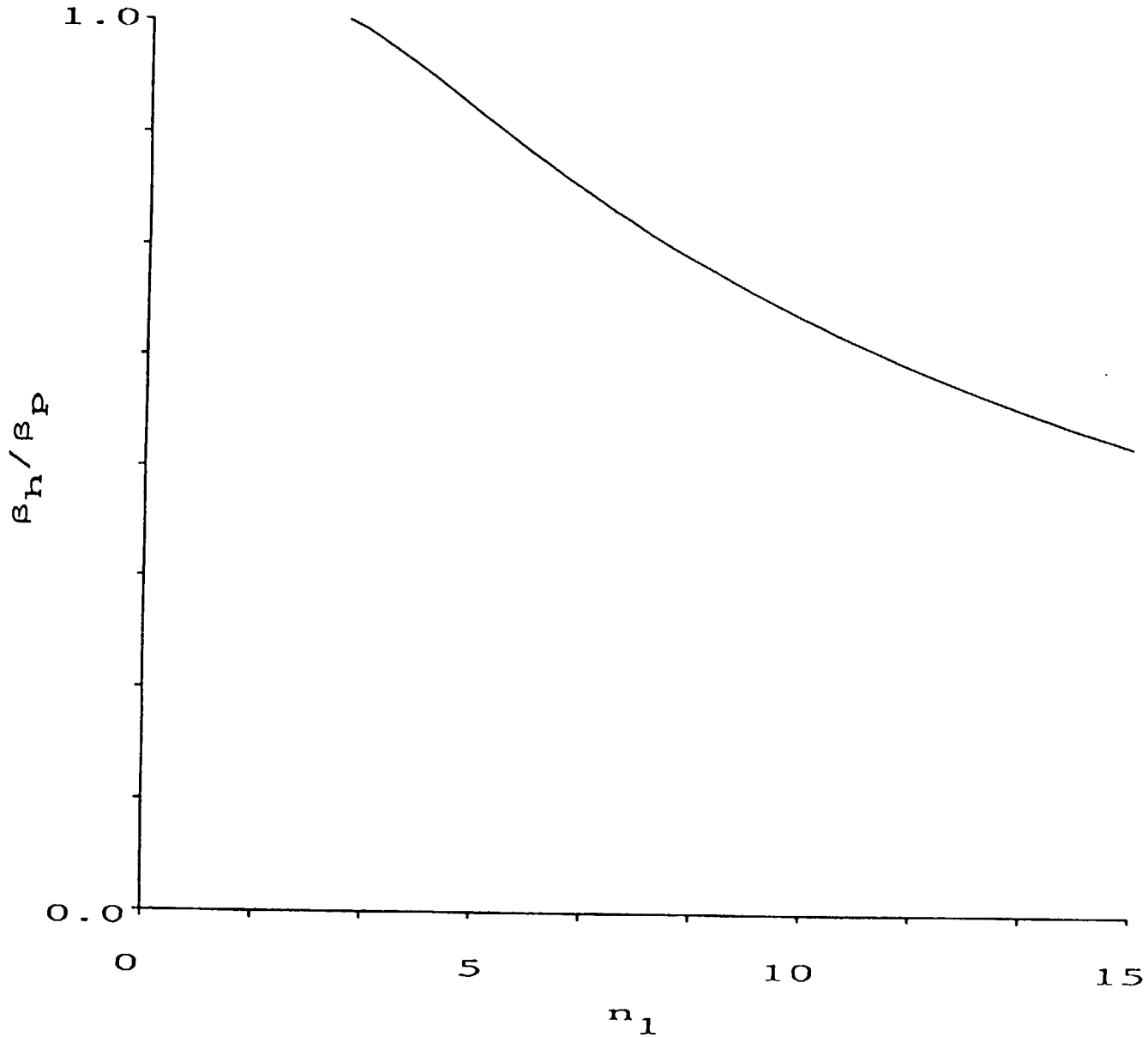


FIGURE 3.20

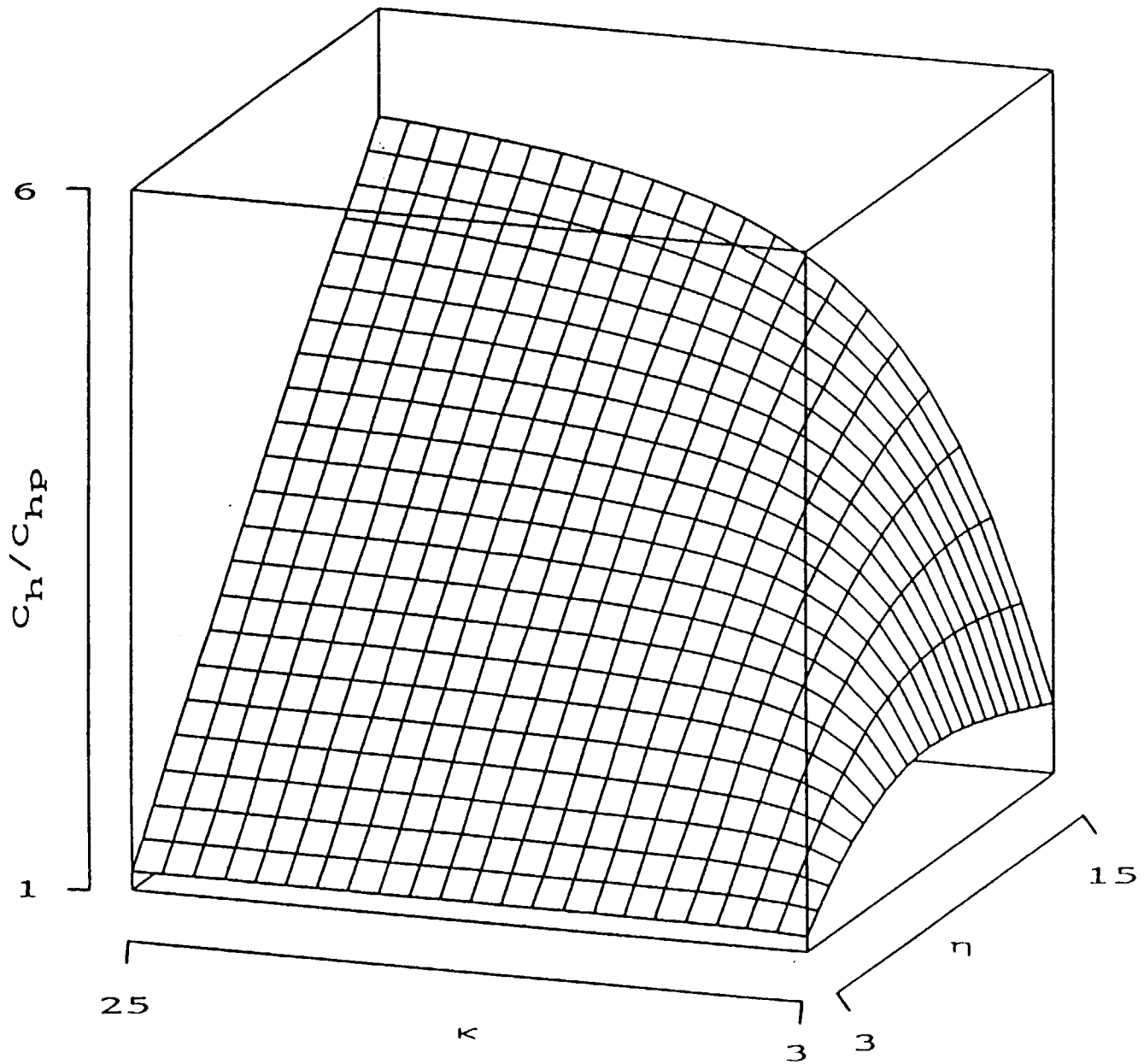


FIGURE 3.21

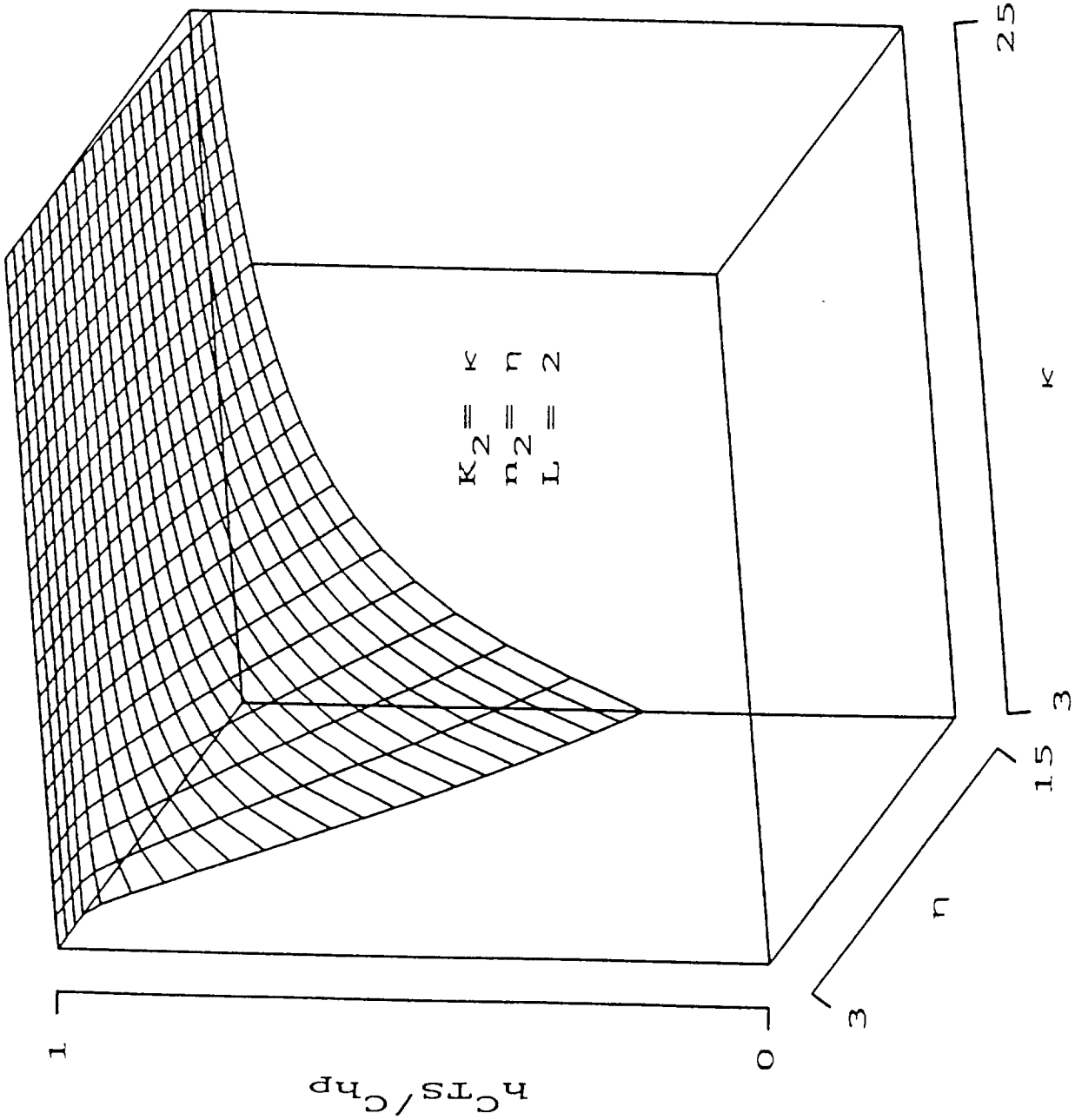


FIGURE 3.22

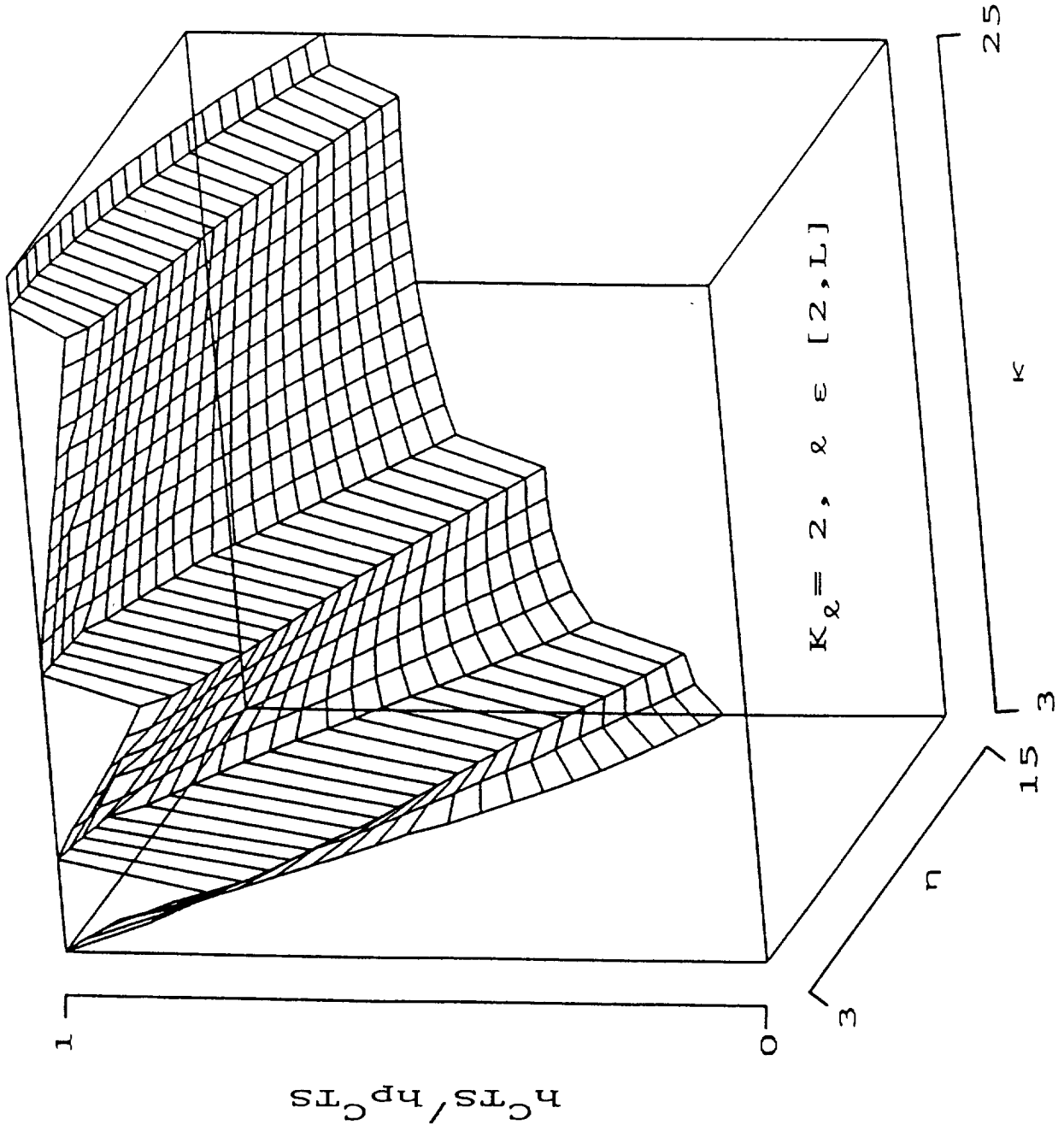


FIGURE 3.23

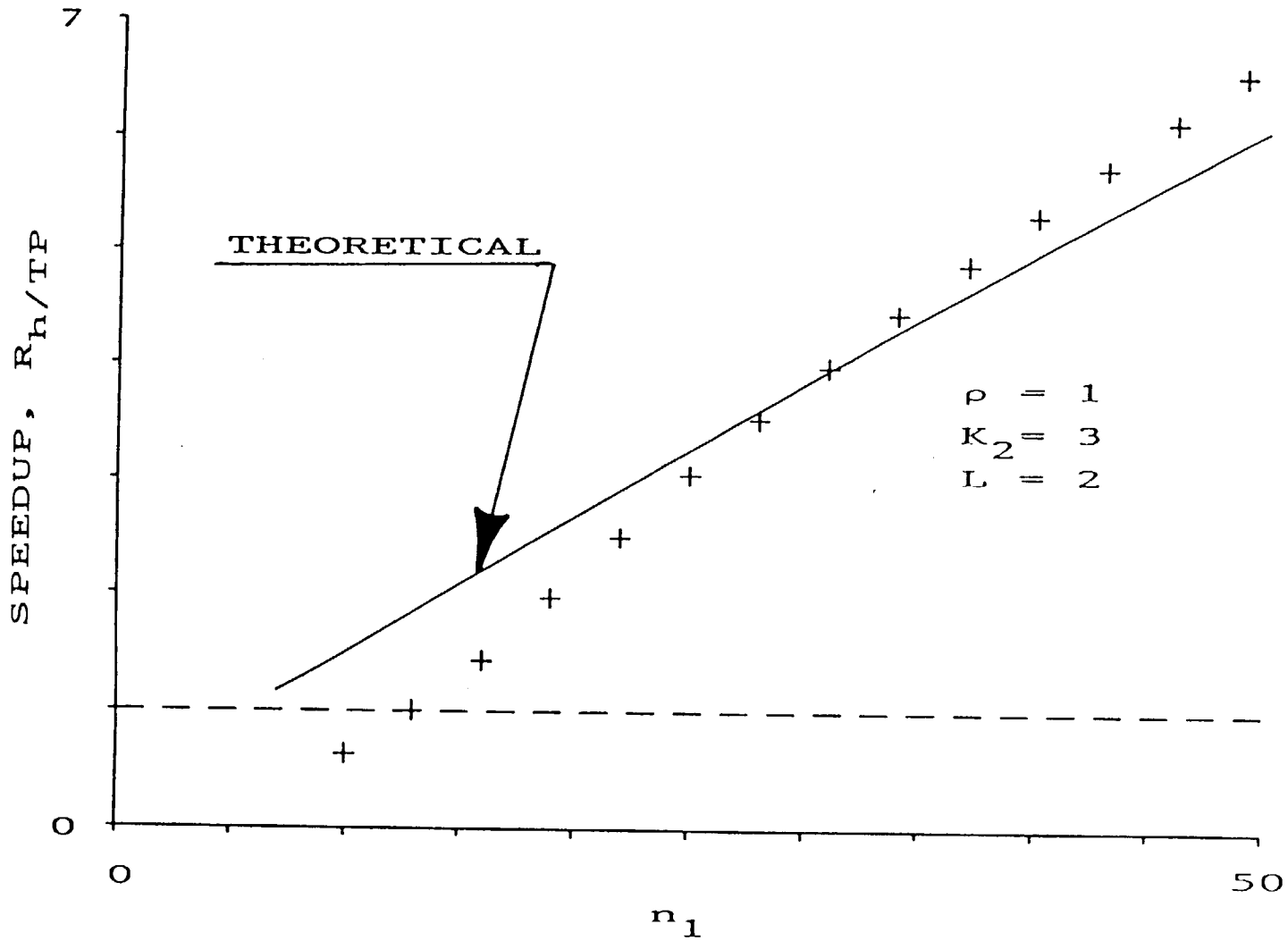


FIGURE 4.1

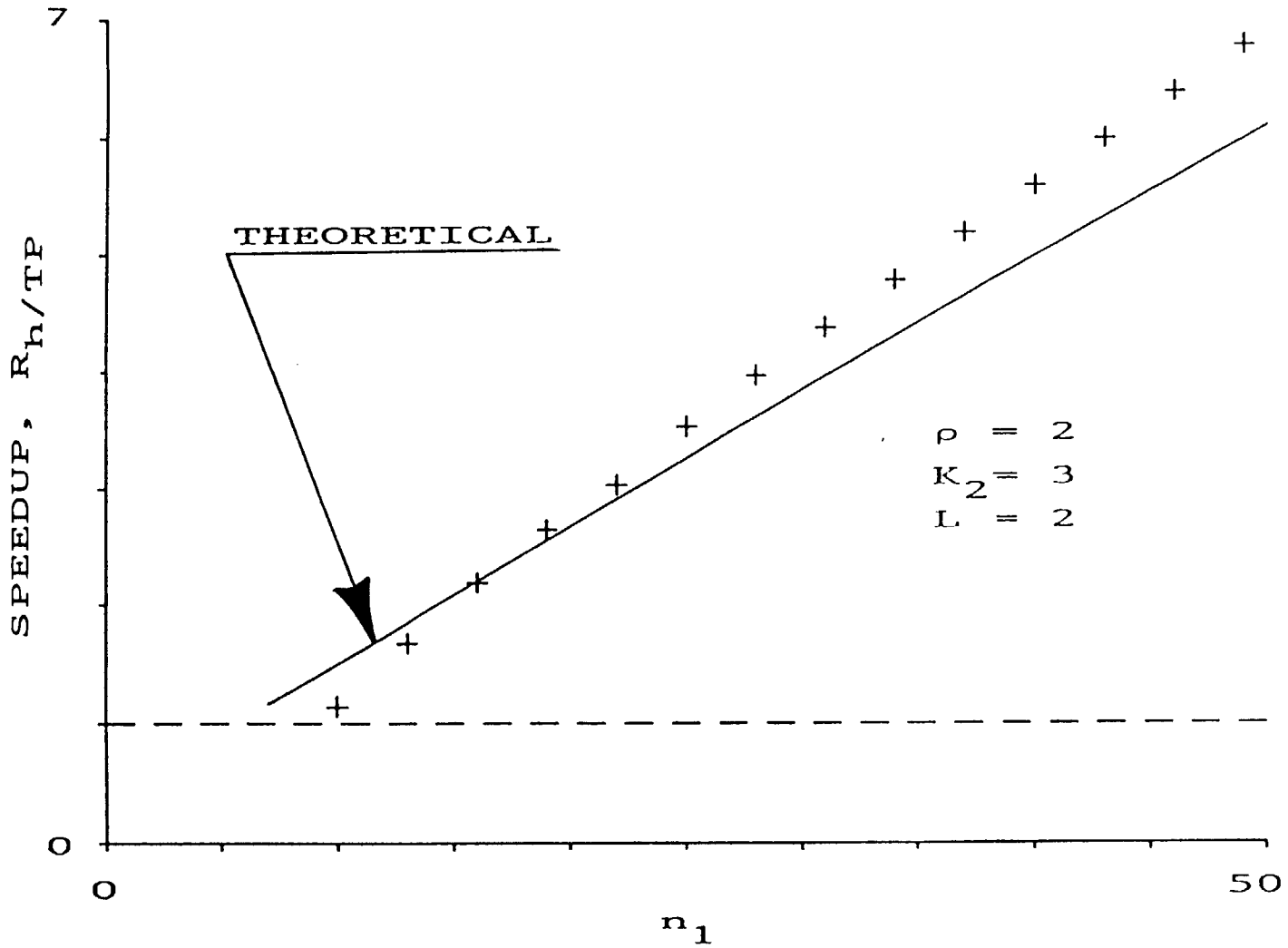


FIGURE 4.2

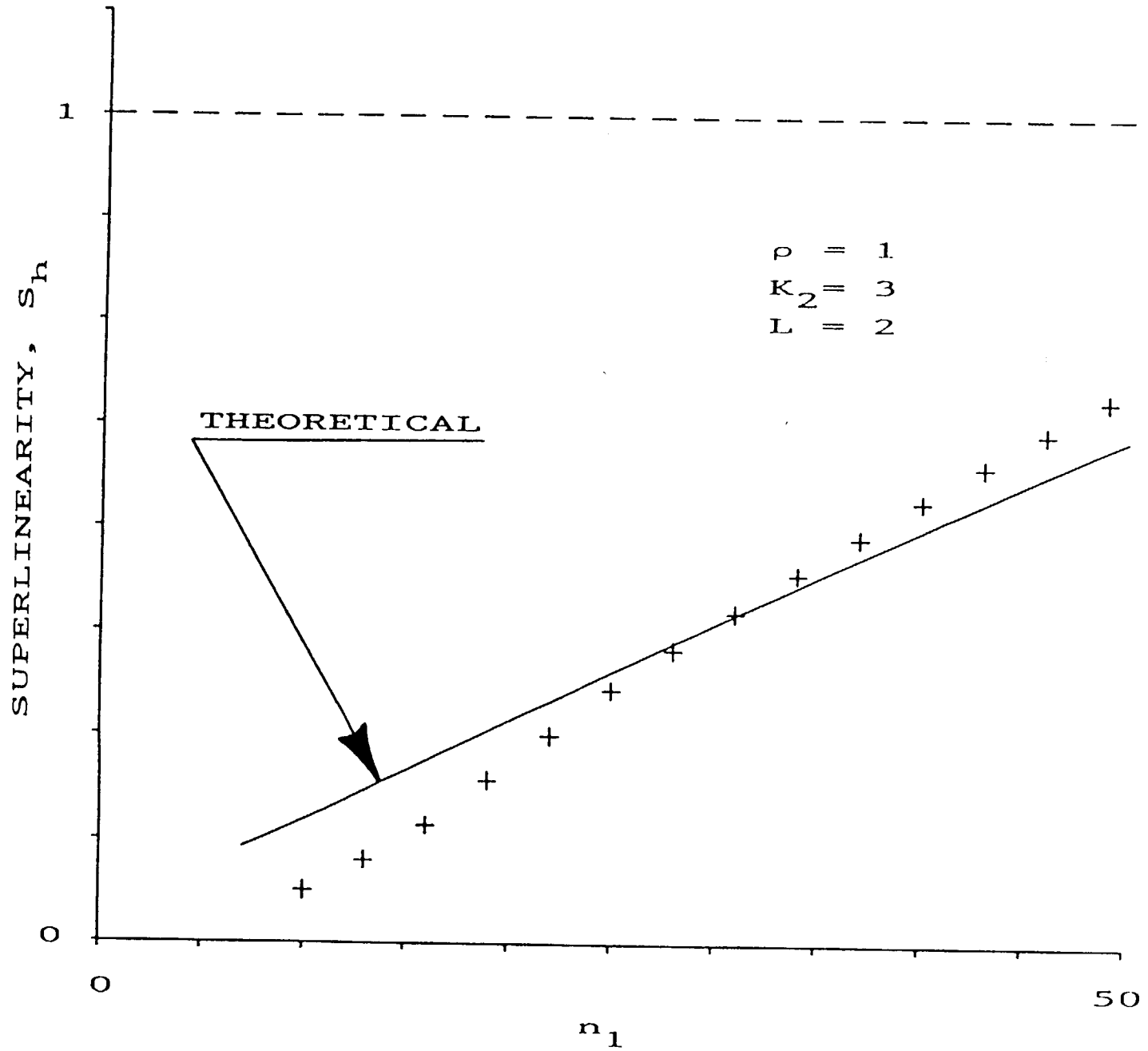


FIGURE 4.3

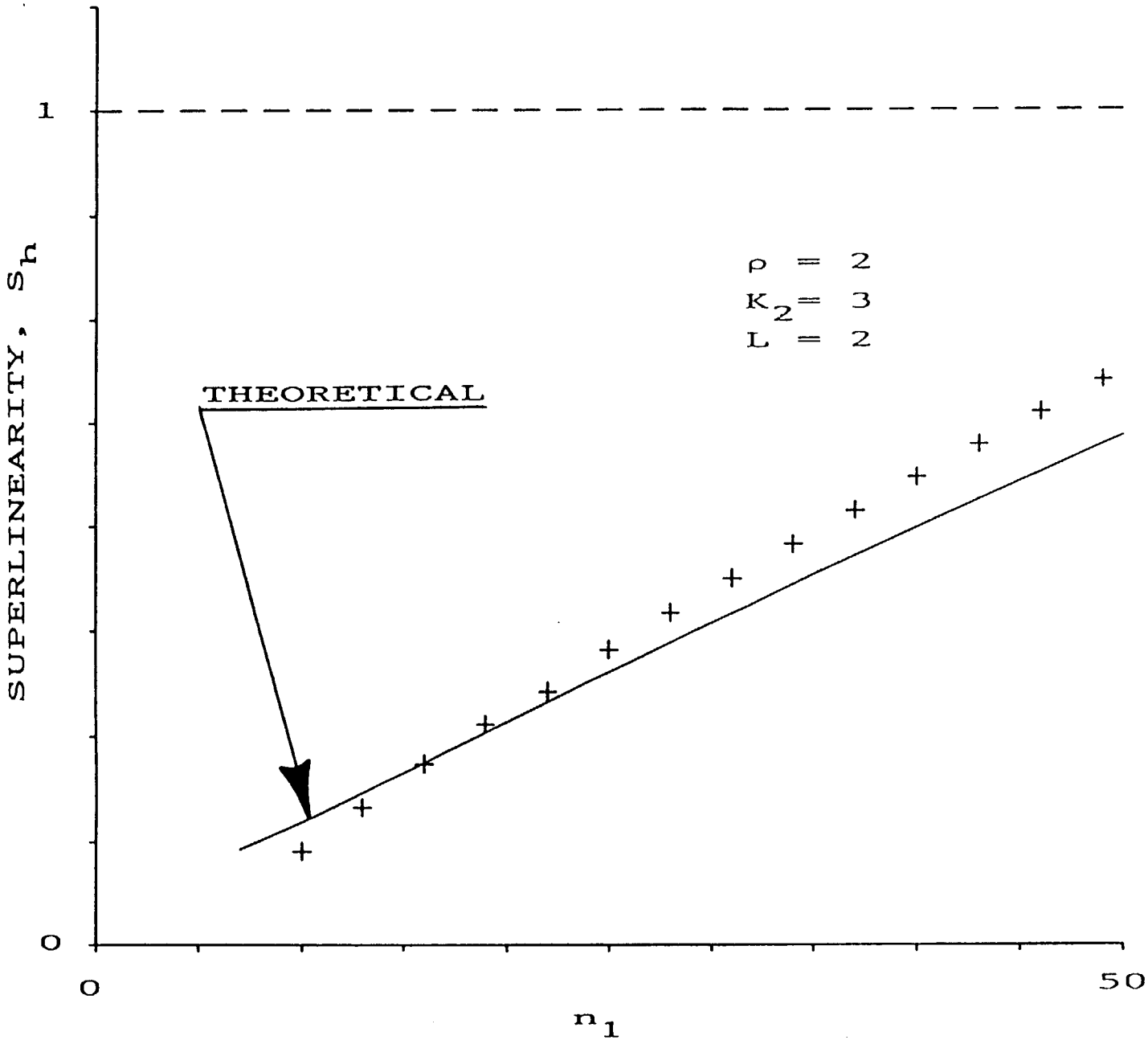


FIGURE 4.4

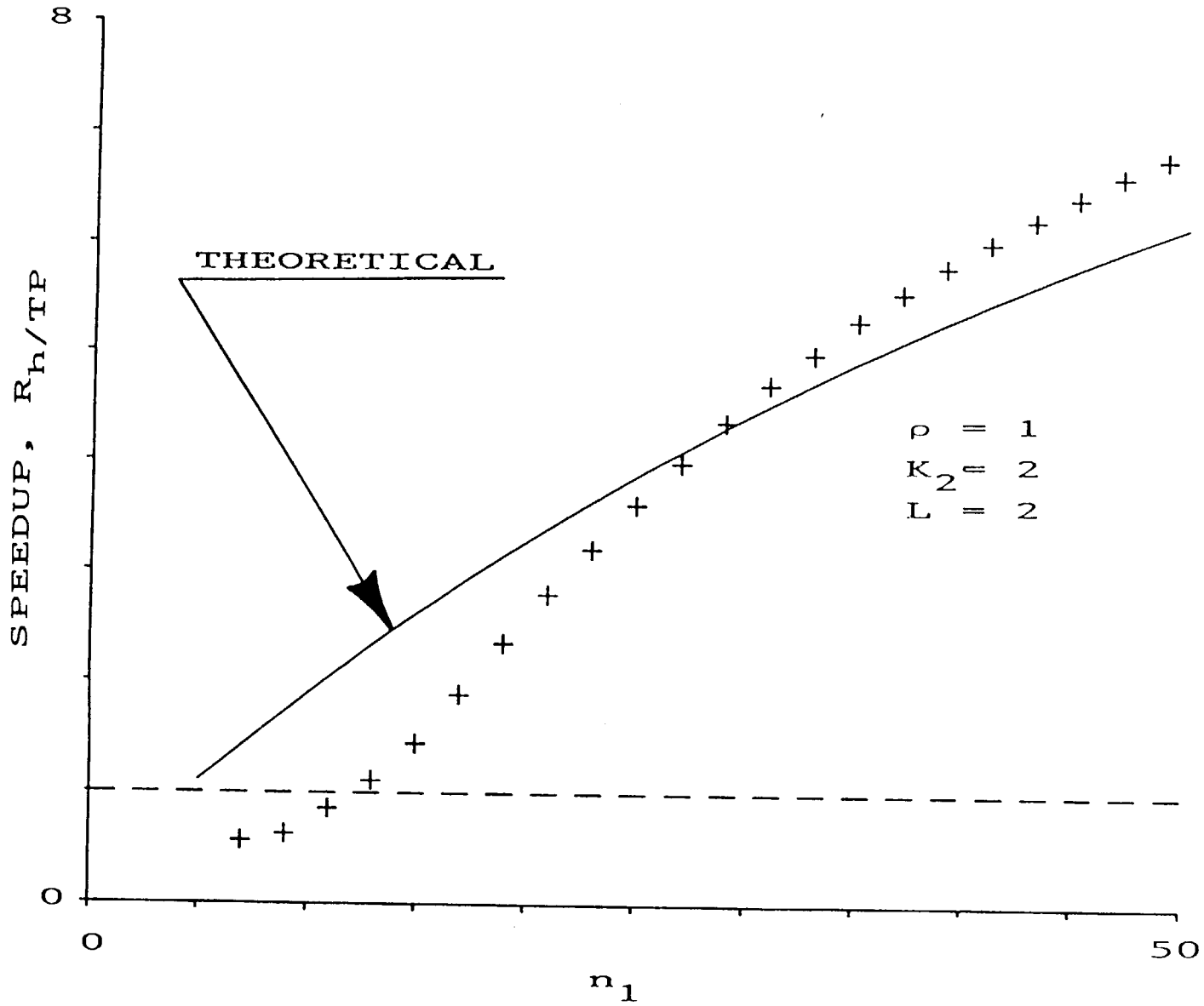


FIGURE 4.5

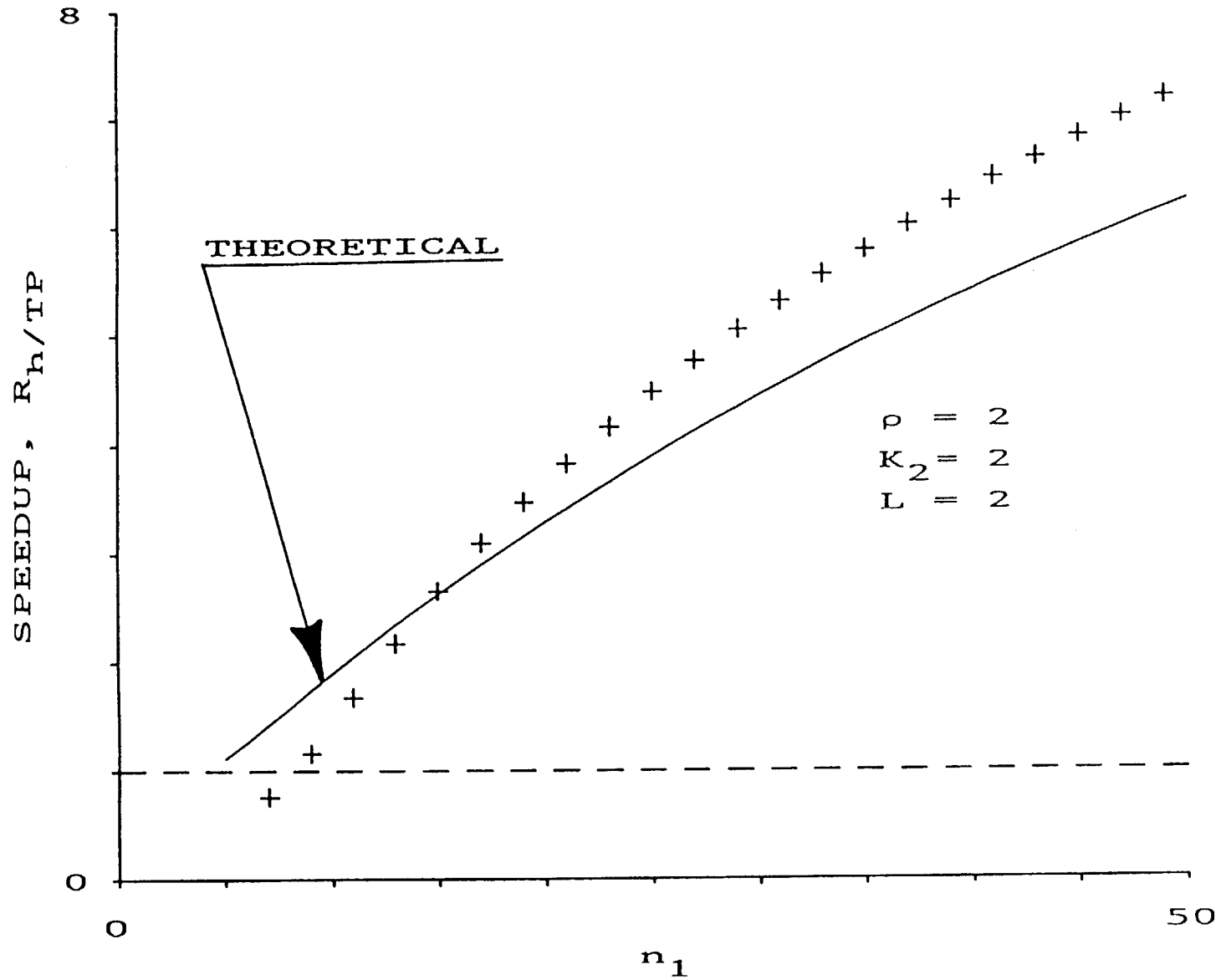


FIGURE 4.6

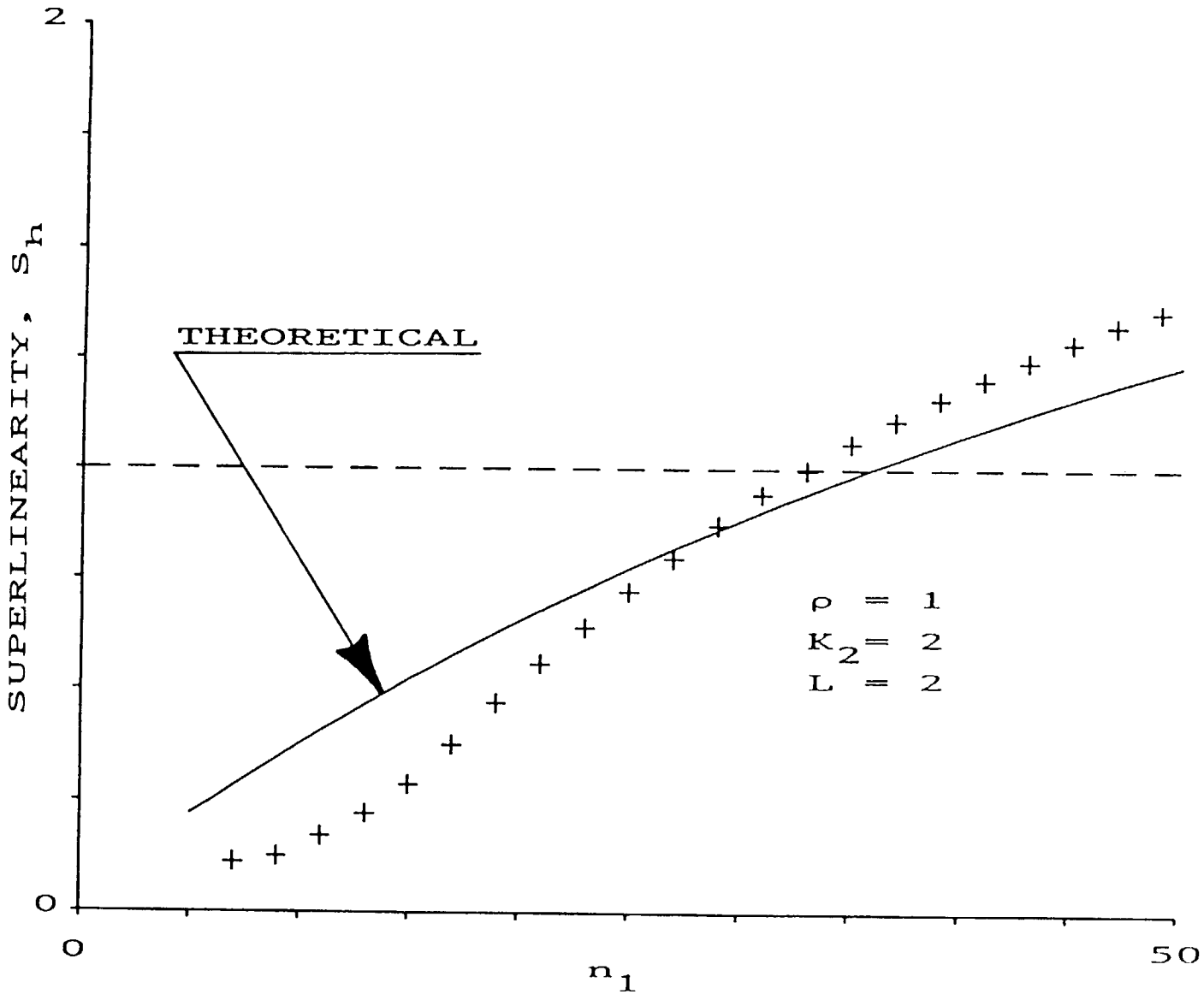


FIGURE 4.7

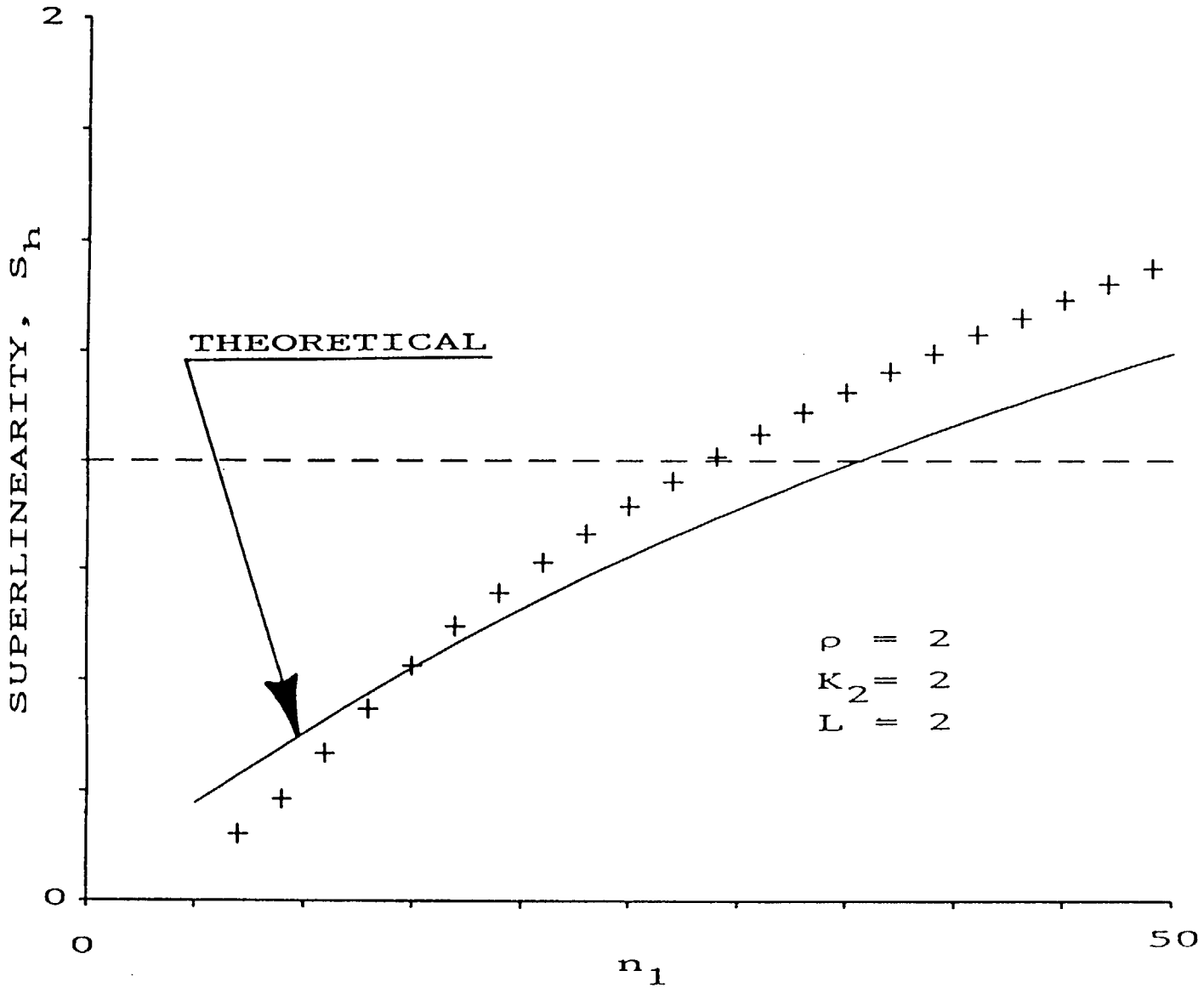


FIGURE 4.8

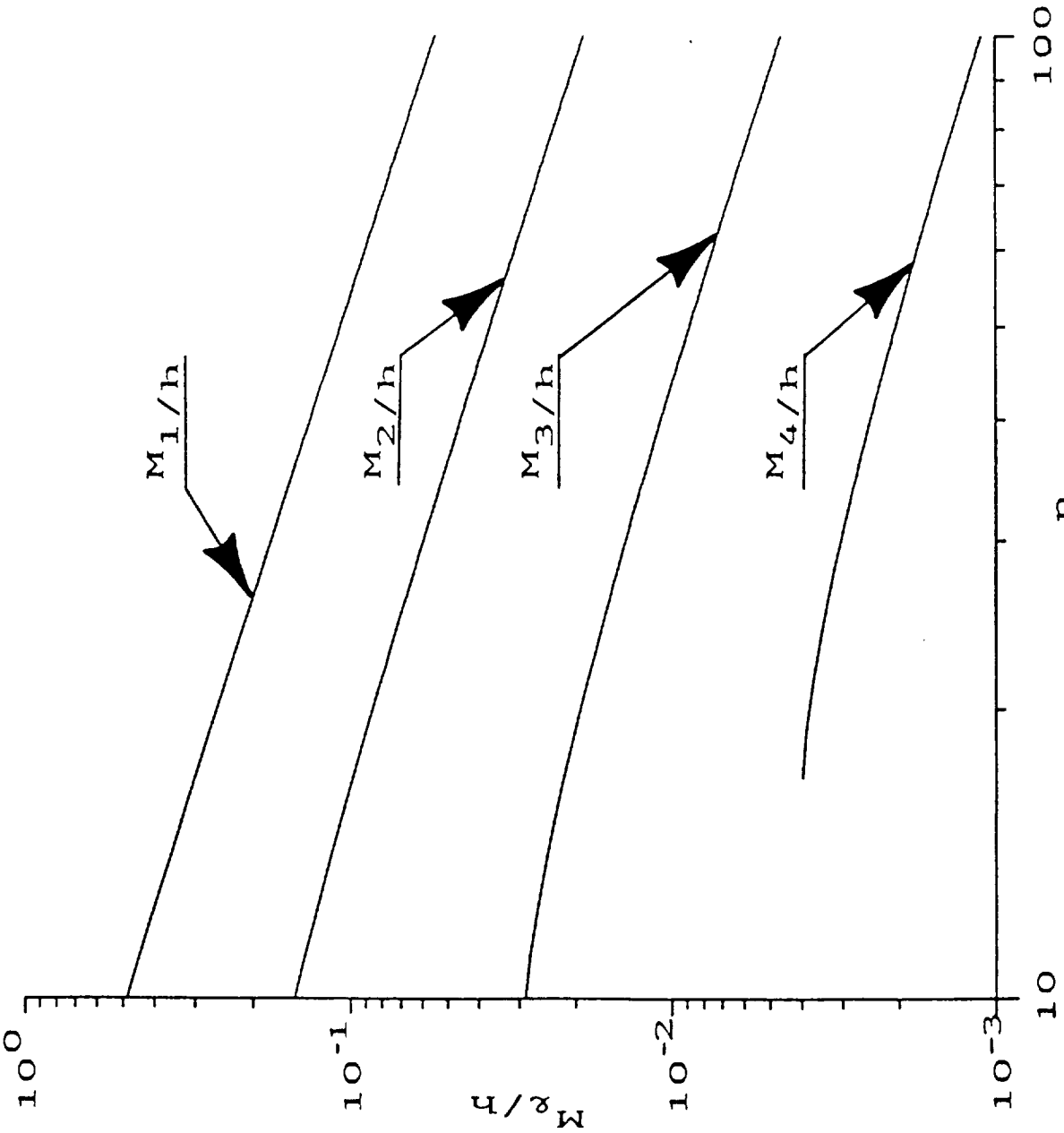


FIGURE 4.9

C-2

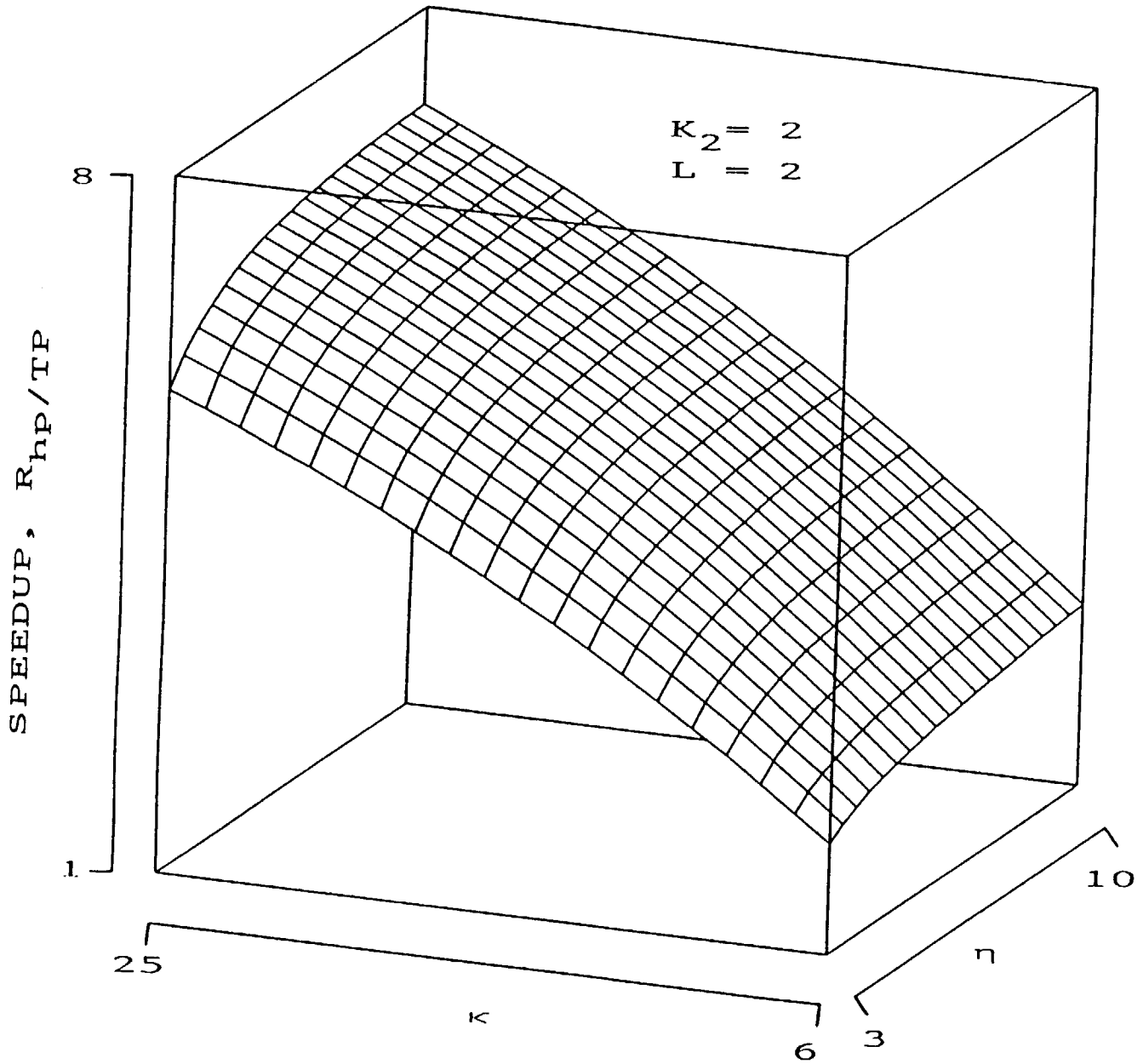


FIGURE 4.10

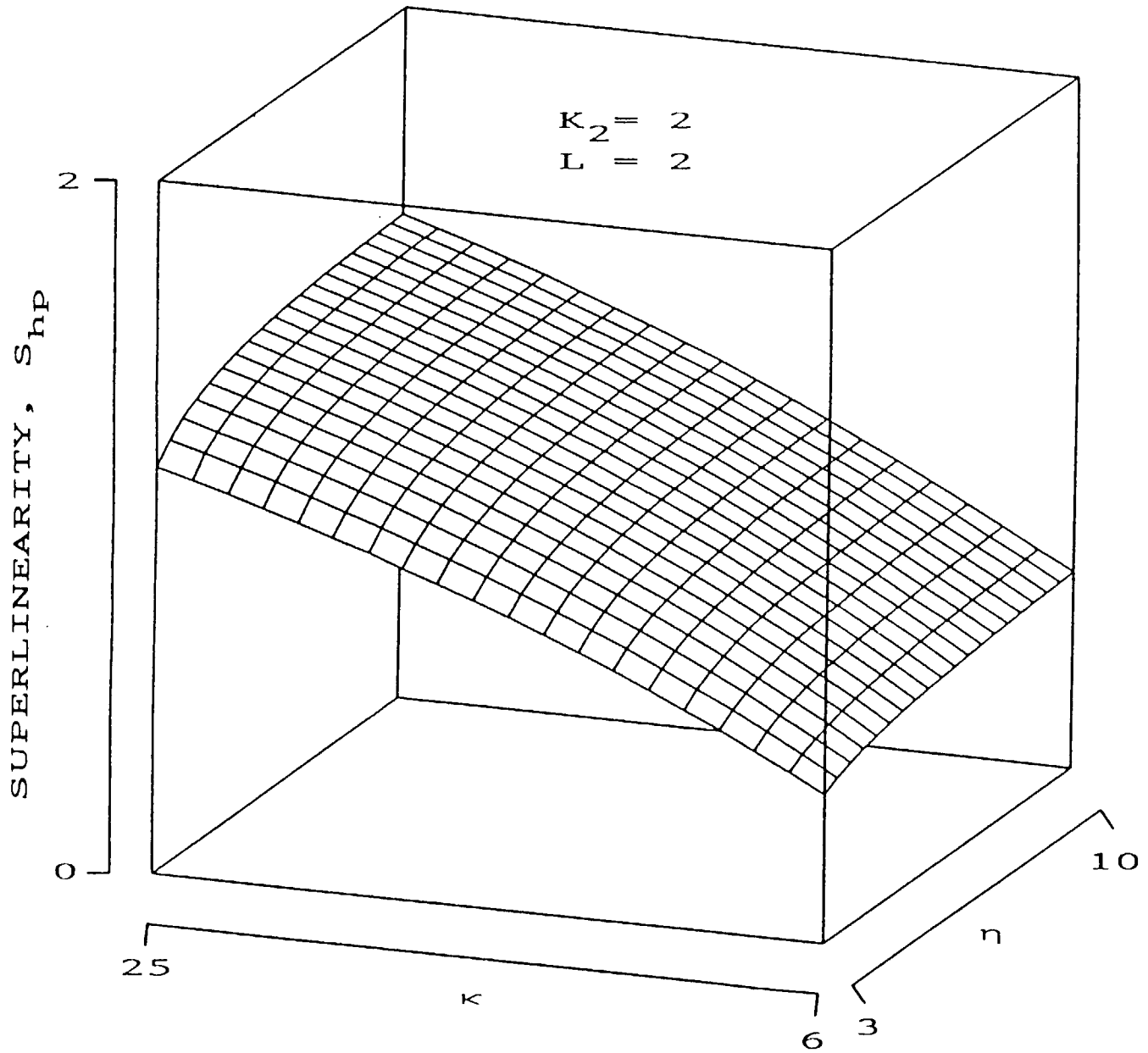


FIGURE 4.11

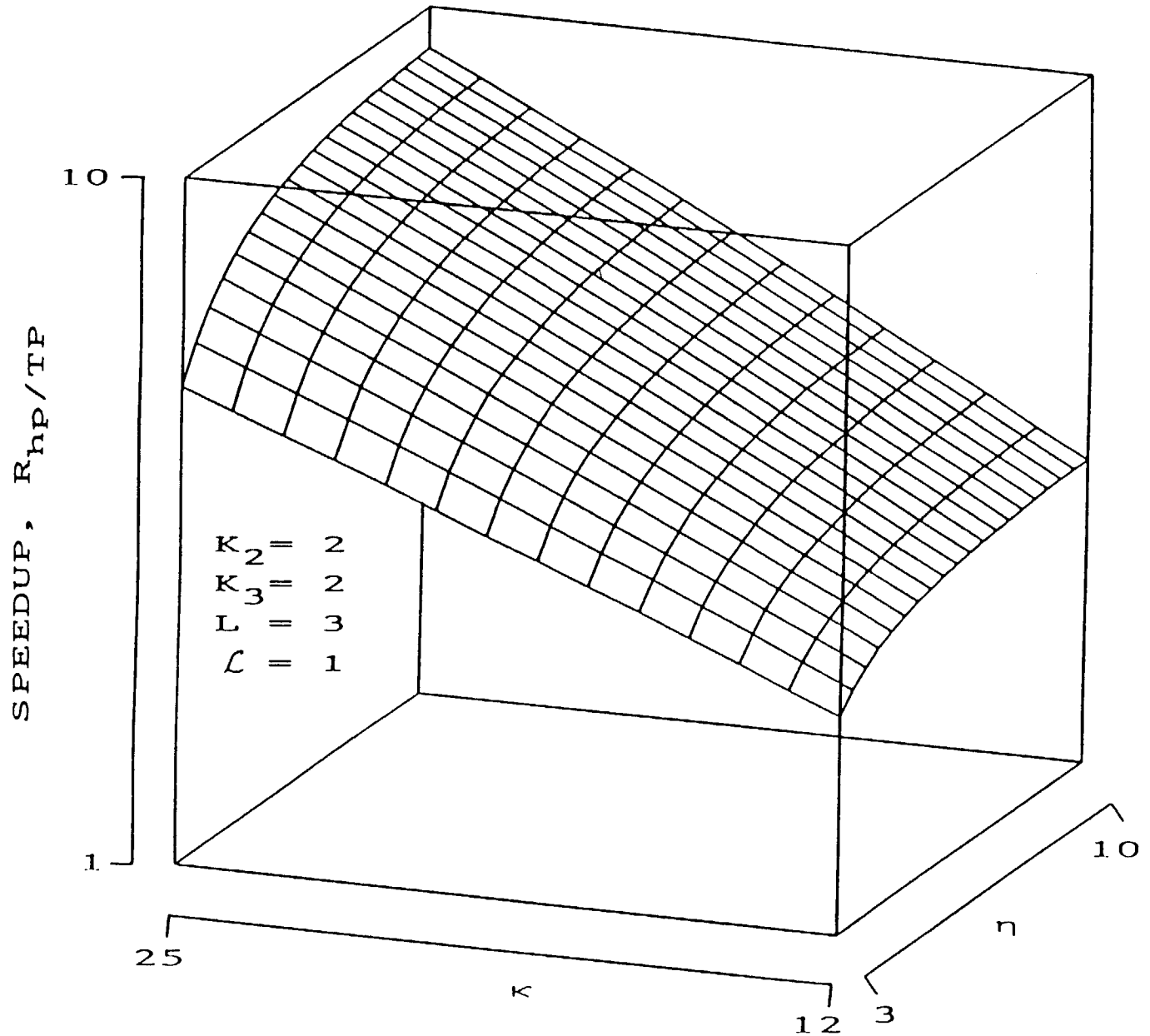


FIGURE 4.12

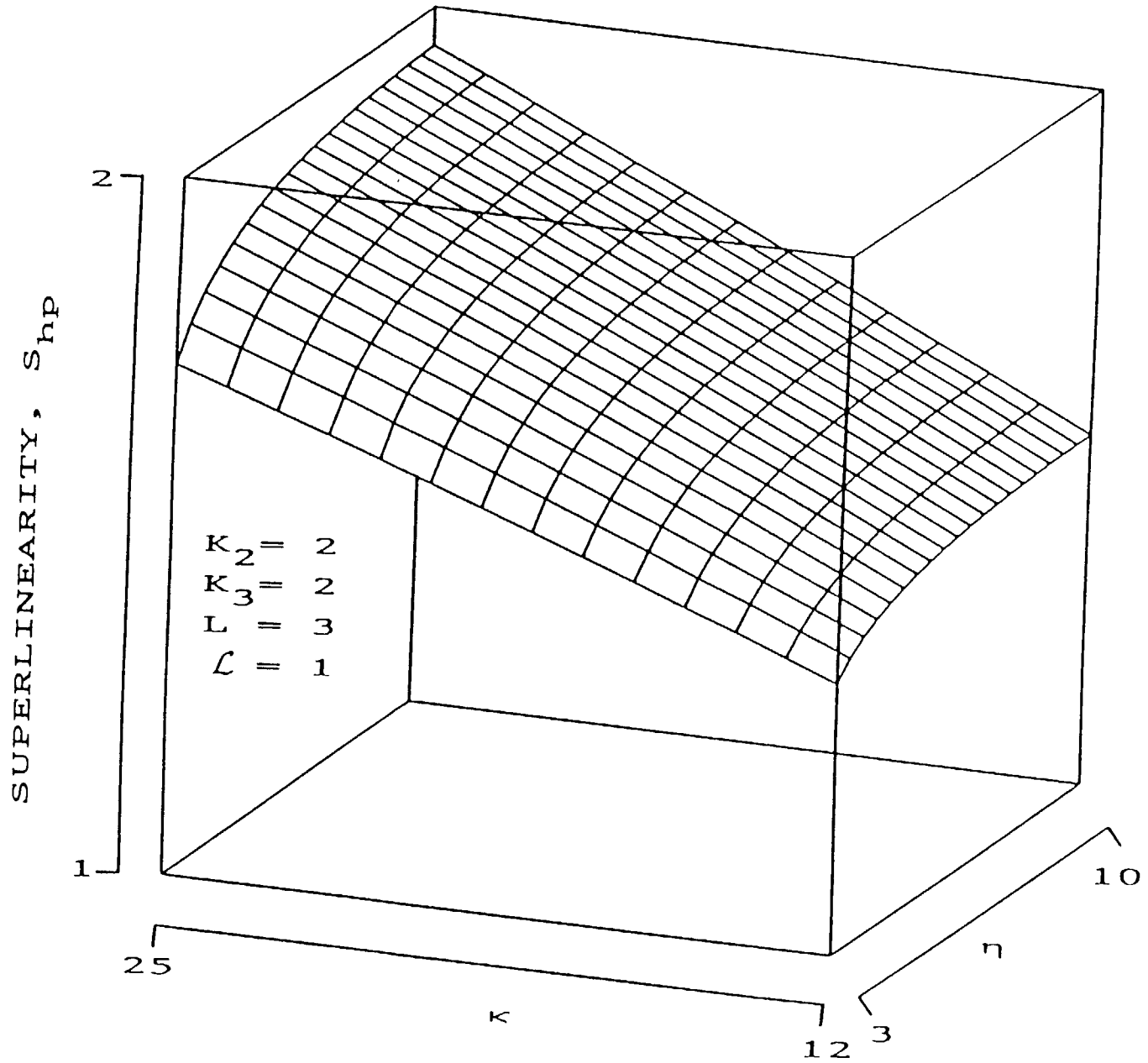


FIGURE 4.13

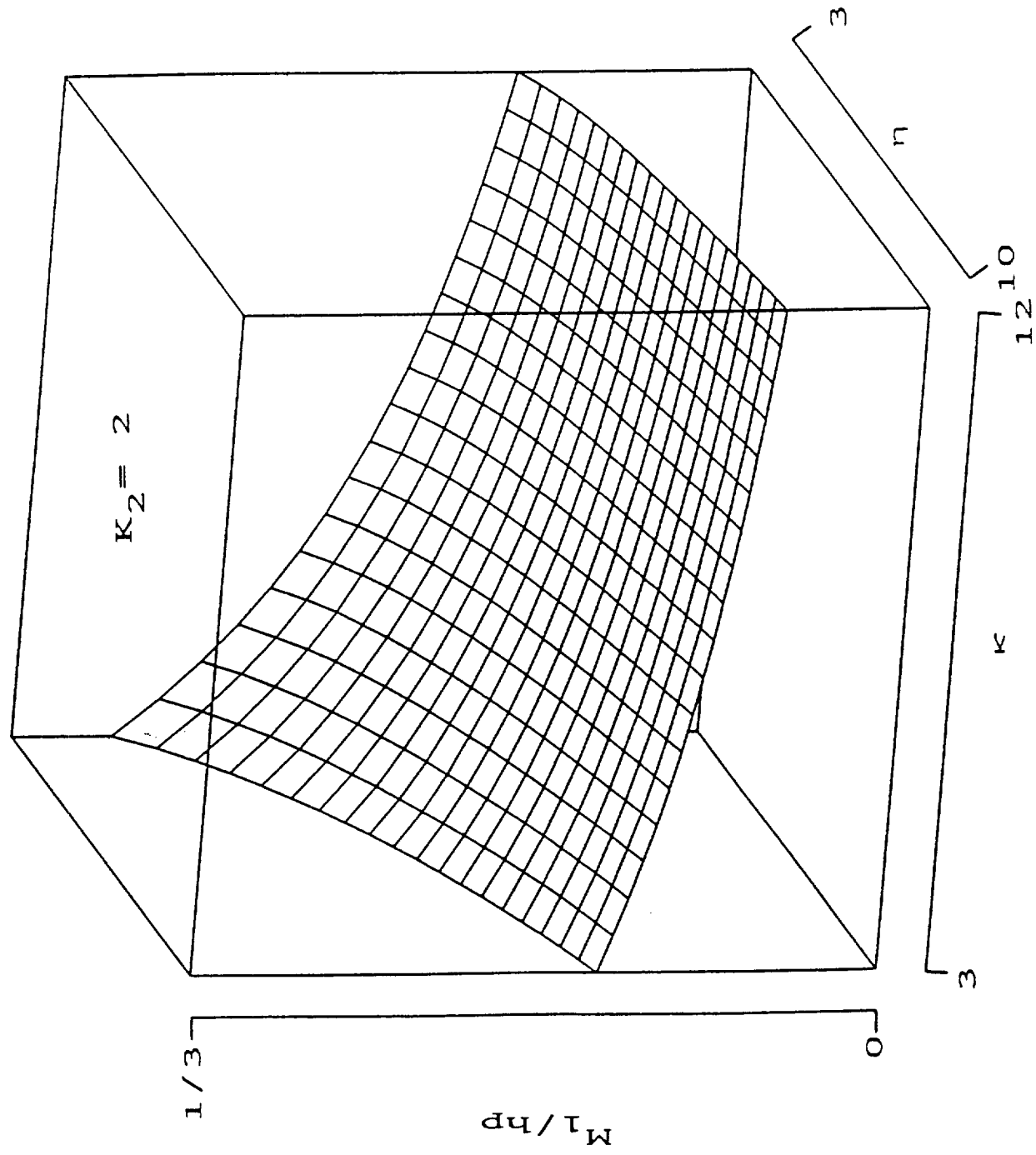
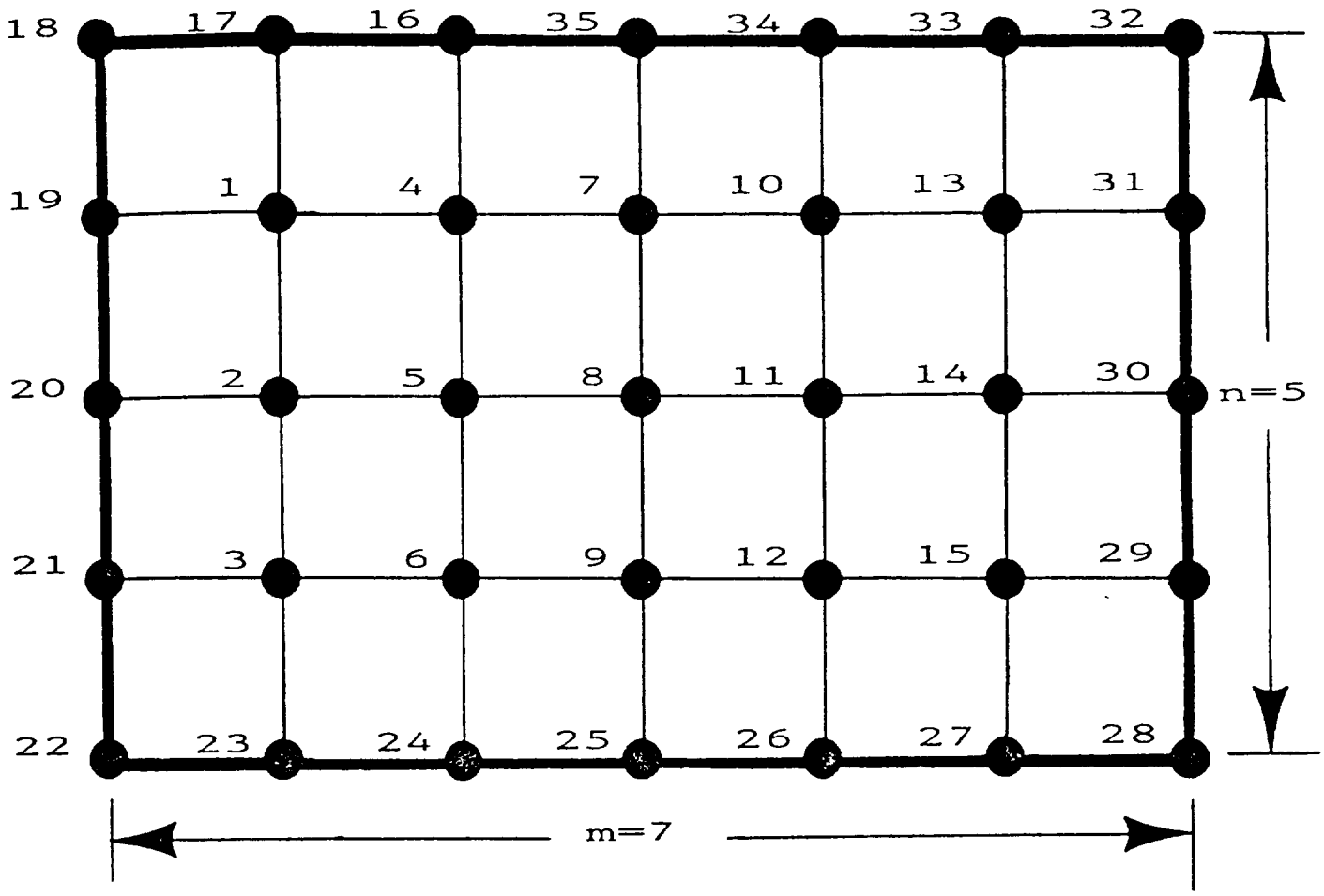


FIGURE 4.14



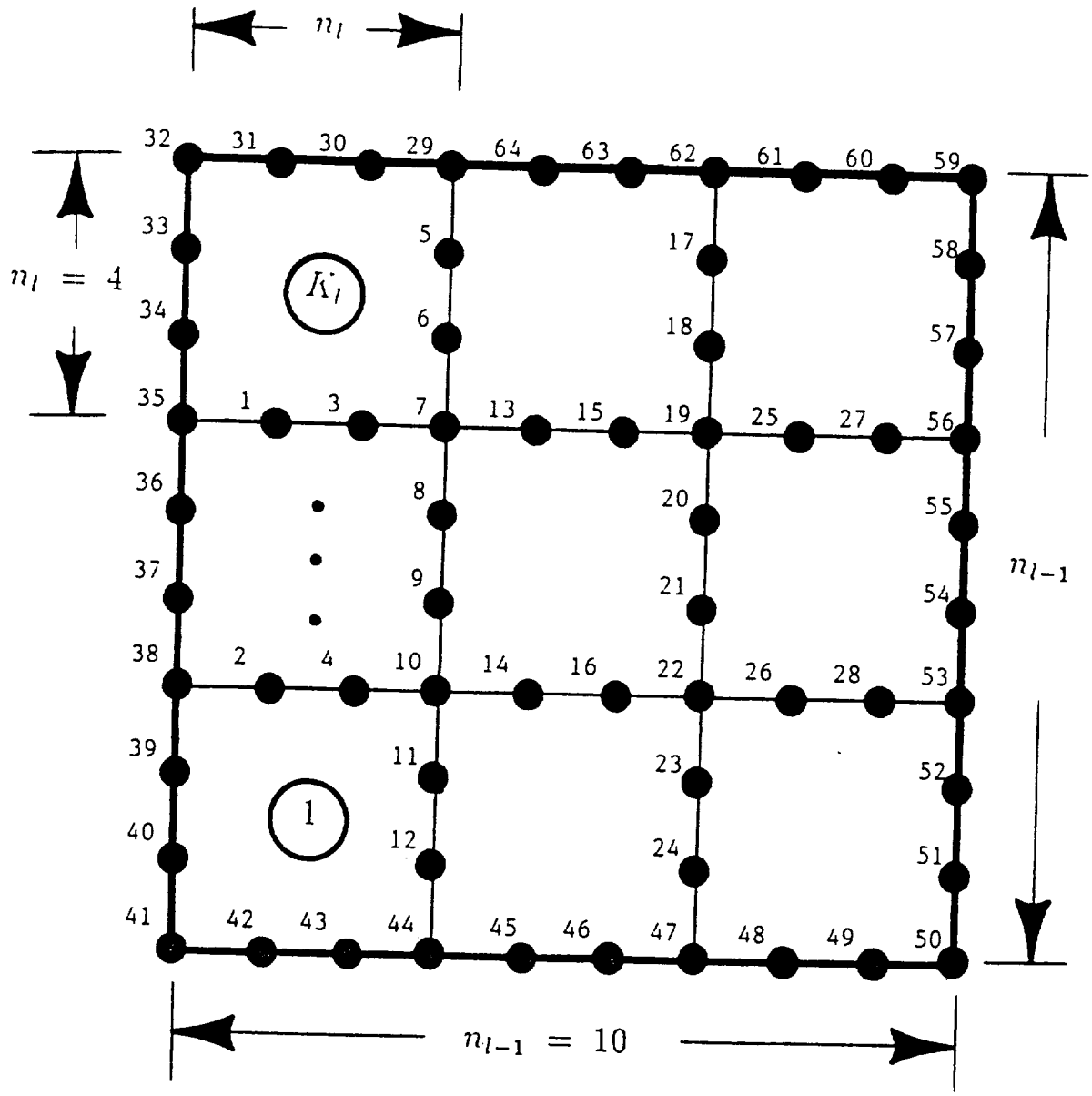
Total Number Of DOF:

$$\alpha = \rho\, m\, n$$

Total Number Of Stiffness Matrix Elements:

$$\beta = \frac{\rho}{2}\left[(2\,m^2\,n + 4\,m\,n^2 - 5\,m\,n - 6\,n^2 - 6\,m + 12\,n)\,\rho + m\,n\right]$$

FIGURE A.1



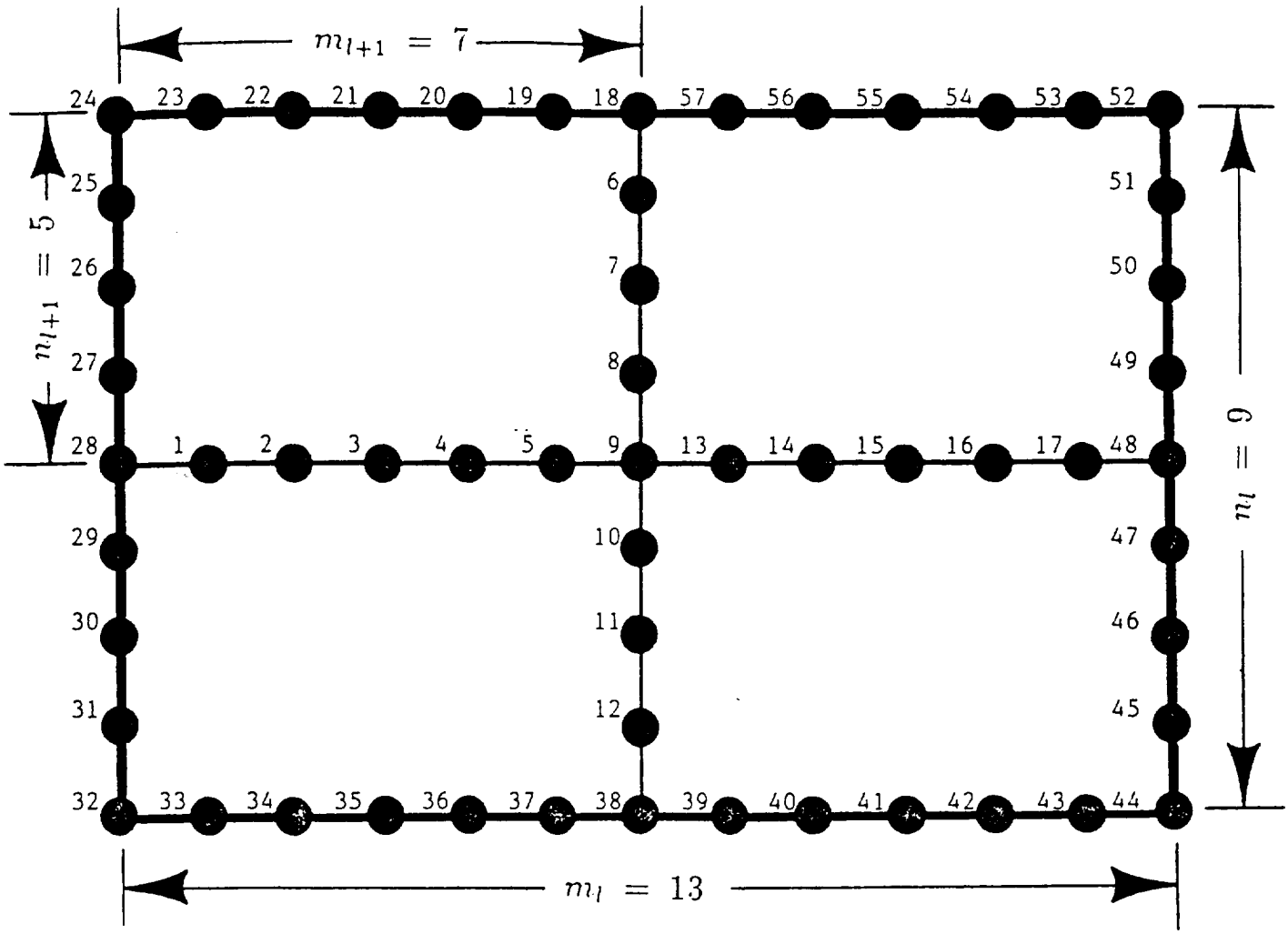
Total Number Of DOF:

$$\alpha = \{2[(K_l)^2 + K_l]n_l - [3(K_l)^2 + 2K_l - 1]\}\rho$$

Total Number Of Stiffness Matrix Elements:

$$\begin{aligned}\beta = \frac{\rho}{2}\{&[14(K_l)^3 + 7(K_l)^2 - 5K_l]\rho(n_l)^2 \\ &- \{[37(K_l)^3 + 13(K_l)^2 - 18K_l]\rho - [2(K_l)^2 + 2K_l]\}n_l \\ &+ \{[24(K_l)^3 + 5(K_l)^2 - 14K_l + 1]\rho - [3(K_l)^2 + 2K_l - 1]\}\}\end{aligned}$$

FIGURE A.2



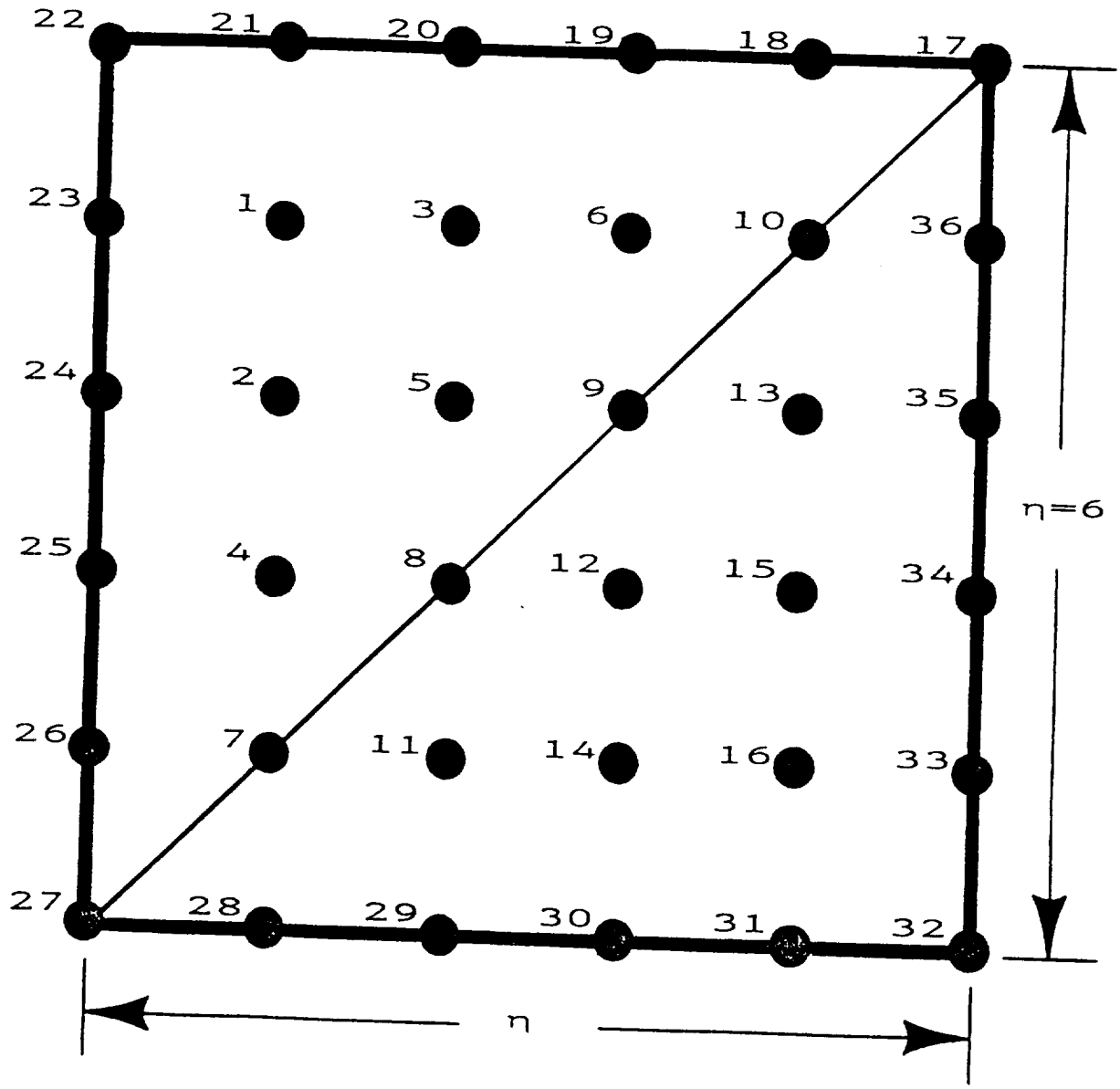
Total Number Of DOF:

$$\alpha = \rho\,(3\,m_l + 3\,n_l - 9)$$

Total Number Of Stiffness Matrix Elements:

$$\beta = \frac{\rho}{4}\{[15(m_l)^2 + 33 m_l n_l + 17(n_l)^2 - 87 m_l - 95 n_l + 123]\rho + [6 m_l + 6 n_l - 18]\}$$

FIGURE A.3



Total Number Of DOF:

$$\alpha = \rho\,\eta^2$$

Total Number Of Stiffness Matrix Elements:

$$\beta = \frac{\rho}{4}\left\{\left[(\eta)^3 + 6\,(\eta)^2 - 11\,\eta + 6\right]\rho + 2\,\eta\right\}\eta$$

FIGURE A.4

PROBLEM SIZE n_1	SECOND LEVEL SUBSTRUCTURE SIZE n_2	$K_2 = 2,\ L = 2$			
		$\rho = 1$		$\rho = 2$	
		THEORETICAL SPEEDUP R_h/TS	ACTUAL SPEEDUP R_h/TS	THEORETICAL SPEEDUP R_h/TS	ACTUAL SPEEDUP R_h/TS
7	4	1.04	0.23	1.07	0.34
9	5	1.23	0.27	1.26	0.45
11	6	1.39	0.37	1.43	0.88
13	7	1.54	0.50	1.57	1.21
15	8	1.67	0.67	1.71	1.50
17	9	1.79	0.90	1.83	1.74
19	10	1.90	1.13	1.93	1.93
21	11	1.99	1.35	2.03	2.09
23	12	2.08	1.57	2.12	2.22
25	13	2.16	1.77	2.20	2.34
27	14	2.24	1.94	2.28	2.43
29	15	2.31	2.10	2.34	2.52
31	16	2.37	2.24	2.41	2.60
33	17	2.43	2.34	2.47	2.67
35	18	2.48	2.45	2.52	2.72
37	19	2.53	2.53	2.57	2.78
39	20	2.58	2.61	2.62	2.83
41	21	2.63	2.67	2.66	2.89
43	22	2.67	2.72	2.71	2.92
45	23	2.71	2.78	2.74	2.97
47	24	2.75	2.83	2.78	3.00
49	25	2.78	2.85	2.82	3.04

TABLE 3.1

PROBLEM SIZE n_1	SECOND LEVEL SUBSTRUCTURE SIZE n_2	$K_2 = K_3 = 2$, $L = 3$			
		$\rho = 1$		$\rho = 2$	
		THEORETICAL SPEEDUP R_h/TS	ACTUAL SPEEDUP R_h/TS	THEORETICAL SPEEDUP R_h/TS	ACTUAL SPEEDUP R_h/TS
10	4	1.22	0.16	1.25	0.43
13	5	1.48	0.27	1.51	0.84
16	6	1.72	0.47	1.75	1.29
19	7	1.94	0.75	1.98	1.71
22	8	2.16	1.08	2.20	2.08
25	9	2.36	1.47	2.40	2.38
28	10	2.55	1.82	2.60	2.66
31	11	2.73	2.13	2.78	2.90
34	12	2.91	2.45	2.95	3.11
37	13	3.07	2.73	3.11	3.32
40	14	3.22	3.00	3.27	3.50
43	15	3.37	3.23	3.42	3.68
46	16	3.51	3.44	3.56	3.83
49	17	3.64	3.62	3.69	3.99

TABLE 3.2

			$K_2 = K_3 = 2,\ L = 3$			
			$\rho = 1$		$\rho = 2$	
PROBLEM SIZE n_1	SECOND LEVEL SUBSTRUCTURE SIZE n_2	THIRD LEVEL SUBSTRUCTURE SIZE n_3	THEORETICAL SPEEDUP R_h/TS	ACTUAL SPEEDUP R_h/TS	THEORETICAL SPEEDUP R_h/TS	ACTUAL SPEEDUP R_h/TS
13	7	4	1.57	0.14	1.62	0.56
17	9	5	1.97	0.31	2.03	1.21
21	11	6	2.36	0.62	2.42	1.93
25	13	7	2.72	1.05	2.79	2.62
29	15	8	3.07	1.56	3.14	3.24
33	17	9	3.40	2.16	3.47	3.78
37	19	10	3.71	2.77	3.79	4.24
41	21	11	4.01	3.31	4.09	4.67
45	23	12	4.29	3.83	4.37	5.06
49	25	13	4.56	4.30	4.65	5.43

TABLE 3.3

	$p = 1,\ L = 2$							
	$K_2 = 2$				$K_2 = 3$			
	THEORETICAL		"ACTUAL"		THEORETICAL		"ACTUAL"	
PROBLEM SIZE n_1	SPEEDUP R_h/TP	SUPER-LINEARITY S_h	SPEEDUP R_h/TP	SUPER-LINEARITY S_h	SPEEDUP R_h/TP	SUPER-LINEARITY S_h	SPEEDUP R_h/TP	SUPER-LINEARITY S_h
7	1.41	0.28	0.56	0.11	1.16	0.12		
13	2.32	0.46	1.11	0.22	1.84	0.18	1.01	0.10
19	3.14	0.63	2.34	0.47	2.55	0.26	1.97	0.20
25	3.87	0.77	3.60	0.72	3.26	0.33	3.04	0.30
31	4.51	0.90	4.71	0.94	3.95	0.40	3.97	0.40
37	5.10	1.02	5.55	1.11	4.64	0.46	4.88	0.49
43	5.63	1.13	6.23	1.25	5.31	0.53	5.73	0.57
49	6.10	1.22	6.79	1.36	5.97	0.60	6.54	0.65
55	6.54	1.31	7.32	1.46	6.62	0.66	7.33	0.73
61	6.94	1.39	7.77	1.55	7.26	0.73	8.09	0.81
67	7.31	1.46	8.16	1.63	7.89	0.79	8.83	0.88
73	7.65	1.53	8.52	1.70	8.51	0.85	9.56	0.96
79	7.96	1.59	8.84	1.77	9.12	0.91	10.26	1.03
85	8.25	1.65	9.14	1.83	9.72	0.97	10.96	1.10
91	8.52	1.70	9.41	1.88	10.31	1.03	11.64	1.16

TABLE 4.1

	ρ = 2, L = 2							
	$K_2 = 2$				$K_2 = 3$			
	THEORETICAL		"ACTUAL"		THEORETICAL		"ACTUAL"	
PROBLEM SIZE n_1	SPEEDUP R_h/TP	SUPER-LINEARITY S_h	SPEEDUP R_h/TP	SUPER-LINEARITY S_h	SPEEDUP R_h/TP	SUPER-LINEARITY S_h	SPEEDUP R_h/TP	SUPER-LINEARITY S_h
7	1.42	0.28	0.76	0.15	1.16	0.12	0.67	0.07
13	2.34	0.47	2.17	0.43	1.84	0.18	1.67	0.17
19	3.16	0.63	3.47	0.69	2.55	0.26	2.65	0.27
25	3.89	0.78	4.47	0.89	3.26	0.33	3.53	0.35
31	4.54	0.91	5.30	1.06	3.96	0.40	4.37	0.44
37	5.13	1.03	6.01	1.20	4.64	0.46	5.19	0.52
43	5.66	1.13	6.62	1.32	5.32	0.53	5.98	0.60
49	6.14	1.23	7.18	1.44	5.98	0.60	6.77	0.68
55	6.58	1.32	7.67	1.53	6.63	0.66	7.54	0.75
61	6.98	1.40	8.11	1.62	7.27	0.73	8.29	0.83
67	7.35	1.47	8.50	1.70	7.91	0.79	9.03	0.90
73	7.69	1.54	8.86	1.77	8.53	0.85	9.76	0.98
79	8.00	1.60	9.19	1.84	9.13	0.91	10.47	1.05
85	8.29	1.66	9.49	1.90	9.73	0.97	11.17	1.12
91	8.56	1.71	9.77	1.95	10.32	1.03	11.86	1.19

TABLE 4.2

$K_2 = K_3 = 2,\ L = 3$								
	$\rho = 1$				$\rho = 2$			
	THEORETICAL		"ACTUAL"		THEORETICAL		"ACTUAL"	
PROBLEM SIZE n_1	SPEEDUP R_h/TP	SUPER-LINEARITY S_h	SPEEDUP R_h/TP	SUPER-LINEARITY S_h	SPEEDUP R_h/TP	SUPER-LINEARITY S_h	SPEEDUP R_h/TP	SUPER-LINEARITY S_h
13	2.45	0.12	0.84	0.04	2.46	0.12	2.00	0.10
17	3.15	0.15	1.54	0.07	3.16	0.15	3.21	0.15
21	3.85	0.18	2.63	0.13	3.86	0.18	4.30	0.20
25	4.55	0.22	3.73	0.18	4.56	0.22	5.35	0.25
29	5.24	0.25	4.89	0.23	5.26	0.25	6.36	0.30
33	5.93	0.28	6.05	0.29	5.95	0.28	7.33	0.35
37	6.62	0.32	7.18	0.34	6.64	0.32	8.28	0.39
41	7.30	0.35	8.23	0.39	7.32	0.35	9.22	0.44
45	7.98	0.38	9.21	0.44	8.00	0.38	10.15	0.48
49	8.66	0.41	10.17	0.48	8.68	0.41	11.08	0.53
53	9.33	0.44	11.13	0.53	9.36	0.45	11.99	0.57
57	10.00	0.48	12.05	0.57	10.03	0.48	12.90	0.61
61	10.67	0.51	12.96	0.62	10.69	0.51	13.80	0.66
65	11.33	0.54	13.82	0.66	11.36	0.54	14.70	0.70
69	11.99	0.57	14.69	0.70	12.02	0.57	15.58	0.74
73	12.65	0.60	15.54	0.74	12.68	0.60	16.46	0.78
77	13.30	0.63	16.37	0.78	13.33	0.63	17.34	0.83
81	13.95	0.66	17.20	0.82	13.98	0.67	18.21	0.87
85	14.59	0.70	18.01	0.86	14.62	0.70	19.07	0.91
89	15.24	0.73	18.82	0.90	15.27	0.73	19.94	0.95
93	15.87	0.76	19.62	0.93	15.91	0.76	20.79	0.99

TABLE 4.3

	$K_2 = K_3 = 2$, L = 3, $\mathcal{C}$ = 1							
	ρ = 1				ρ = 2			
	THEORETICAL		"ACTUAL"		THEORETICAL		"ACTUAL"	
PROBLEM SIZE n_1	SPEEDUP R_h/TP	SUPER-LINEARITY S_h	SPEEDUP R_h/TP	SUPER-LINEARITY S_h	SPEEDUP R_h/TP	SUPER-LINEARITY S_h	SPEEDUP R_h/TP	SUPER-LINEARITY S_h
13	2.34	0.47	0.46	0.09	2.36	0.47	1.43	0.29
17	2.99	0.60	0.94	0.19	3.02	0.60	2.59	0.52
21	3.63	0.73	1.73	0.35	3.66	0.73	3.69	0.74
25	4.26	0.85	2.67	0.53	4.29	0.86	4.72	0.94
29	4.87	0.97	3.70	0.74	4.91	0.98	5.67	1.13
33	5.47	1.09	4.78	0.96	5.52	1.10	6.55	1.31
37	6.06	1.21	5.82	1.16	6.11	1.22	7.37	1.47
41	6.64	1.33	6.76	1.35	6.69	1.34	8.18	1.64
45	7.21	1.44	7.65	1.53	7.26	1.45	8.94	1.79
49	7.77	1.55	8.49	1.70	7.82	1.56	9.69	1.94
53	8.31	1.66	9.32	1.86	8.37	1.67	10.41	2.08
57	8.85	1.77	10.07	2.01	8.91	1.78	11.12	2.22
61	9.37	1.87	10.81	2.16	9.44	1.89	11.79	2.36
65	9.89	1.98	11.47	2.29	9.96	1.99	12.46	2.49
69	10.39	2.08	12.15	2.43	10.46	2.09	13.10	2.62
73	10.89	2.18	12.78	2.56	10.96	2.19	13.74	2.75
77	11.38	2.28	13.39	2.68	11.45	2.29	14.34	2.87
81	11.86	2.37	13.97	2.79	11.93	2.39	14.96	2.99
85	12.33	2.47	14.54	2.91	12.40	2.48	15.53	3.11
89	12.79	2.56	15.09	3.02	12.87	2.57	16.13	3.23
93	13.24	2.65	15.63	3.13	13.32	2.66	16.68	3.34

TABLE 4.4

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	January 1993	Final Contractor Report

4. TITLE AND SUBTITLE

Hierarchical Poly Tree Configurations for the Solution of Dynamically Refined Finite Element Models

5. FUNDING NUMBERS

WU-505-63-53
NAG3-664

6. AUTHOR(S)

G.D. Gute and J. Padovan

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

The University of Akron
Akron, Ohio 44325

8. PERFORMING ORGANIZATION REPORT NUMBER

E-7532

9. SPONSORING/MONITORING AGENCY NAMES(S) AND ADDRESS(ES)

National Aeronautics and Space Administration
Lewis Research Center
Cleveland, Ohio 44135-3191

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

NASA CR-191054

11. SUPPLEMENTARY NOTES

Project Manager, C.C. Chamis, Structures Division, NASA Lewis Research Center, (216) 433-3252.

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Unclassified - Unlimited
Subject Category 39

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This paper demonstrates how a multilevel substructuring technique, called the Hierarchical Ploy Tree (HPT), can be used to integrate a localized mesh refinement into the original finite element model more efficiently. The optimal HPT configurations for solving isoparametrically square h-, p-, and hp-extensions on single and multiprocessor computers is derived. In addition, the reduced number of stiffness matrix elements that must be stored when employing this type of solution strategy is quantified. Moreover, the HPT inherently provides localize "erro-trapping" and a logical, efficient means with which to isolate physically anomalous and analytically singular behavior.

14. SUBJECT TERMS

Multilevel substructuring; Mesh refinement; H.P. hp-meshes; Parallel processors; Error trapping; Singular behavior

15. NUMBER OF PAGES

112

16. PRICE CODE

A06

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	